Universidad
Carlos III de Madrid

PROYECTO FIN DE CARRERA

Escuela Politécnica Superior
Ingeniería en Informática
Departamento de Informática

Creación de una herramienta para la generación de analizadores esteganográficos para imágenes

JUBSAC (Java Universal Blind StegAnalyzer Creator)

AUTOR: Javier García-Cuerva Velasco
TUTOR: Jorge Blasco Alís

Leganés, Octubre de 2010

Título: Creación de una herramienta para la generación de analizadores esteganográficos para imágenes

Autor: Javier García-Cuerva Velasco

Director: Jorge Blasco Alís

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO ***PRESIDENTE***

AGRADECIMIENTOS

Hay mucha gente a la cual me gustaría agradecer todo su apoyo, pero en especial a ti, porque te lo mereces y por muchas razones que cito a continuación:

- Gracias, por haberme animado a conseguir lo imposible.
- Gracias, por haberme dado la mano cuando me caía.
- Gracias, por acompañarme en el camino.
- Gracias, por haber hecho de mis fracasos grandes éxitos.
- Gracias, por tú gran empatía conmigo.
- Gracias, por haber reído conmigo.
- Gracias, por haber llorado conmigo.
- Gracias, por haberte divertido conmigo.
- Gracias, por ayudarme a sacar fuerzas de flaqueza.
- Gracias, por pasar tanto tiempo conmigo.
- Gracias, por comprenderme tan bien siempre.
- Gracias, por darme todo lo que necesito para conseguir lo imposible.
- Gracias, por darme la energía que a veces me falta.
- Gracias, por hacerme más fuerte cada día.
- Gracias, por enseñarme todo lo que sabes.
- Gracias, por defenderme ante todo.
- Gracias, por darme mucho más de lo que yo te doy.
- Gracias, por hacerme sentir siempre bien.
- Gracias, por guiarme por el camino correcto.
- Gracias, por sacarme del camino erróneo.
- Gracias, por quererme tanto.

Realmente, muchas gracias por todo. No hay precio para devolverte todo lo que has hecho por mí. Te llevaré siempre en mi corazón.

Javier García-Cuerva Velasco.

RESUMEN

La Esteganografía, es la ciencia que estudia las técnicas de ocultación de información u objetos dentro de otros, llamados portadores, de modo que no se perciba su existencia. Es una mezcla de artes y técnicas que se combinan para conformar la práctica de ocultar y enviar información sensible en un portador que pueda pasar desapercibido. El Estegoanálisis, es la ciencia que estudia las técnicas que se usan para detectar y/o anular información oculta por la esteganografía. Al igual que con la esteganografía, es considerada una mezcla entre técnicas y arte para descubrir la información oculta.

El proyecto presentado en la siguiente memoria, trata sobre la realización de un programa capaz de crear analizadores o detectores esteganográficos para imágenes. Estos detectores, también llamados estegoanalizadores, son programas capaces de detectar la presencia de información oculta en archivos digitales de imágenes. Se ha construido un procedimiento, por el cual se puede estegoanalizar cualquier algoritmo esteganográfico a través de una serie de atributos que se extraen de las imágenes. Se ha implementado este esquema procedimental, en una aplicación. Esta aplicación, la cual ha sido llamada JUBSAC, las siglas en inglés de Creador de EstegoAnalizadores Universales a Ciegas en Java (Java Universal Blind StegAnalyzer Creator).

El principal objetivo de JUBSAC es la experimentación e investigación de nuevos métodos estegoanalíticos mediante técnicas de inteligencia artificial. Si bien hay otras propuestas para el estegoanálisis mediante el uso de técnicas de inteligencia artificial, no sé ha abordado hasta la fecha un marco general para la realización de este tipo de procedimientos experimentales.

Para comprobar la eficacia de la herramienta desarrollada se han realizado una serie de experimentos sobre un algoritmo esteganográfico. JUBSAC ha permitido agilizar y simplificar el proceso gracias al uso de la inteligencia artificial.

Se ha utilizado el mejor analizador, con un 100% de eficacia, resultante de los experimentos, para construir una implementación de un estegoanalizador para ser usado directamente desde JUBSAC. Por lo tanto, JUBSAC puede ser usada, además de para la experimentación, para detectar información oculta en imágenes.

Palabras clave: Esteganografía, Estegoanálisis, Estegoanalizador, Esteganográfico, Imagen, Inteligencia Artificial, JUBSAC.

“Un enemigo sólo puede controlar tu información si puede encontrarla”

Peter Wayner

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. INTRODUCCIÓN Y OBJETIVOS

1.1 Introducción

Este proyecto presenta el desarrollo de una aplicación que permite generalizar el proceso de creación de detectores de información oculta en imágenes. Estos detectores son llamados estegoanalizadores y el proyecto se centrará en los estegoanalizadores universales o a ciegas. Son llamados así debido a que, sólo trabajan con el medio portador, es decir, el medio usado para ocultar información, del que se sospecha que contiene información oculta. También son denominados como universales, porque no se posee más información, como la herramienta esteganográfica usada o el medio portador original.

La motivación de desarrollar una aplicación de estas características es la inexistencia de una aplicación semejante. Existen estudios que crean estegoanalizadores, usando una serie de procedimientos, que son comunes entre los distintos estudios. En cambio, estos estudios sólo crean un estegoanalizador para ser usado en el experimento que estén tratando para dicho estudio.

La aplicación desarrollada en el proyecto, va más allá que todos estos estudios, ya que aglutina todas estas técnicas y las generaliza en un marco único. Así, futuros estudios, de distintos autores, podrán realizarse con esta aplicación y poder realizar una comparación de resultados, nunca antes conseguida, entre estudios de orígenes distintos. El marco construido es un procedimiento general, que consta de varias partes para la creación de estegoanalizadores de forma automática. Las partes del procedimiento han sido creadas después de un laborioso trabajo de análisis y estudio de artículos que versan sobre cómo crear un estegoanalizador para imágenes. Debido a la generalización de todas las técnicas en un único procedimiento, la aplicación tiene como objetivo principal la experimentación e investigación de nuevos métodos estegoanalíticos.

Esta aplicación ha sido nombrada JUBSAC, las siglas en inglés de Creador de EstegoAnalizadores Universales a Ciegas en Java (Java Universal Blind StegAnalyzer Creator). JUBSAC, en las partes de las que consta su procedimiento, emplea técnicas como el uso de herramientas esteganográficas, la extracción de medidas en imágenes, o el uso de algoritmos de la Inteligencia Artificial. La combinación de estas técnicas puede crear estegoanalizadores. Como un factor positivo extra, JUBSAC es ampliable. Es decir, cualquier estudioso del estegoanálisis puede añadir sus propios parámetros, como son nuevas herramientas esteganográficas, o nuevas medidas para extraerlas de las imágenes, o nuevos algoritmos de la Inteligencia Artificial.

Finalmente, para comprobar la eficacia de JUBSAC se han realizado una serie de experimentos sobre diferentes algoritmos esteganográficos. Después se ha usado el mejor analizador, resultante de los experimentos, para construir una implementación de un estegoanalizador para ser usado directamente desde JUBSAC. Por lo tanto, JUBSAC puede ser usada tanto para la experimentación como para la posterior detección de imágenes con información oculta.

1.2 Objetivos

El objetivo fundamental del proyecto es el desarrollo de un marco para la generación de estegoanalizadores. Para llegar a tal objetivo se construirá una aplicación que implemente dicho marco siendo así capaz de generalizar el proceso de creación de detectores de información oculta en imágenes.

En base a este objetivo principal, se proponen los siguientes objetivos parciales:

- Abstraer todas las técnicas existentes para el estegoanálisis de imágenes, para desarrollar un esquema o procedimiento único, que generalice la creación de estegoanalizadores de imágenes.
- Implementar el procedimiento desarrollado en una aplicación única.
- Facilitar que la herramienta sea ampliable.
- Permitir que como último paso se puedan obtener modelos que se puedan ejecutar directamente sobre imágenes.

1.3 Estructura de la Memoria

En las siguientes líneas se describe la estructura que sigue esta memoria.

- **Capítulo 1, Introducción:** Introduce el proyecto, sus objetivos y la estructura de la memoria.

- **Capítulo 2, Conceptos Básicos:** Explica todos los conceptos teóricos relacionados con la esteganografía y el estegoanálisis, para que el lector no tenga problemas para entender el resto del documento.

- **Capítulo 3, Análisis:** Describe el estado del arte con las diferentes técnicas documentadas sobre la esteganografía, el estegoanálisis, la extracción de medidas de imágenes, y la Inteligencia Artificial. Finalmente, se describe el procedimiento sobre cómo se pretende obtener un estegoanalizador para cualquier algoritmo esteganográfico y se hace un análisis de las funcionalidades de la creación de estegoanalizadores.

- **Capítulo 4, Diseño**: Explica el diseño de la aplicación que implementa la creación de estegoanalizadores.

- **Capítulo 5, Implementación:** Especifica los detalles de la implementación del creador de analizadores esteganográficos.

- **Capítulo 6, Experimentación:** Específica los diferentes experimentos realizados y muestra el desarrollo de un estegoanalizador en base a los mejores resultados obtenidos en los experimentos.

- **Capítulo 7, Conclusiones y Líneas Futuras:** Analiza las conclusiones finales después de haber finalizado el proyecto junto a una**s** ampliaciones y posibles mejoras de la herramienta.

- **Capítulo 8, Gestión del Proyecto:** Especifica las fases del desarrollo del proyecto, los medios empleados, la planificación del proyecto y el presupuesto de la realización del proyecto descrito en esta memoria.

- **Capítulo 9, Glosario:** Enumera la correspondencia de las siglas y acrónimos citados en el texto de la presente memoria.

- **Capítulo 10, Referencias Usadas:** Lista las referencias usadas durante la memoria, a artículos y herramientas software usados para la realización del proyecto.

- **Anexo 1, Manual del Usuario**: Manual para usar JUBSAC.

- **Anexo 2, Manual del Programador:** Manual para ampliar JUBSAC.

2. CONCEPTOS BÁSICOS

2.1 Introducción

Antes de centrarse en los aspectos más específicos del proyecto desarrollado, se explicarán unos conceptos claves para entender mejor el posterior texto. Los términos en inglés estarán acompañados siempre de la correspondiente traducción al castellano.

Durante el resto del documento se pueden encontrar dos términos frecuentemente:

- **Esteganografía (*Steganography*)**
- **Estegoanálisis (*Steganalysis*)**

Dichas palabras no se encuentran contempladas en la vigésima segunda edición del diccionario de la Real Academia Española, pero sí son usadas en el argot informático y de la seguridad de la información, a nivel mundial. No confundir esteganografía con estenografía, que es sinónimo de taquigrafía, y estas dos palabras si están contempladas en el diccionario.

- **Esteganografía:** Del griego "*steganos*" (oculto) y "*graphein*" (escribir). Lo que podría traducirse como escritura oculta. Es la disciplina que estudia técnicas que permiten ocultar o camuflar datos dentro de otros, llamados portadores, de modo que no se perciba su existencia.
- **Estegoanálisis**: Son las técnicas que se usan para detectar mensajes ocultos mediante la esteganografía. Es la detección (ataques pasivos) y/o anulación (ataques activos) de información oculta en distintas cubiertas (por ejemplo las imágenes digitales), así como la posibilidad de localizar la información útil dentro de la misma (existencia y tamaño).

Existe una analogía, entre esteganografía y estegoanálisis, y los términos criptografía y criptoanálisis. Pero sus objetivos son distintos, ya que la criptografía lo que pretende es cifrar información para que esta sea ininteligible por terceras personas, cuando existe un canal de comunicación entre un emisor y un receptor. Pero aunque un tercer invitado, no deseado, en la vía comunicativa, no pueda entender que información está siendo intercambiada, si dispone de una información con bastante valor. Esta información es el conocimiento del hecho de que hay un emisor y un receptor que están intercambiando información lo suficientemente importante como para ir cifrada. Por ende, dicho conocimiento valioso es imposible de ocultar sólo con los métodos criptográficos. Es aquí donde entra en juego la esteganografía, cuya función principal es que no haya conocimiento de que existe comunicación entre dos partes, yendo esta información cifrada o no. En la Figura 1 se muestra la diferencia entre criptografía y esteganografía:

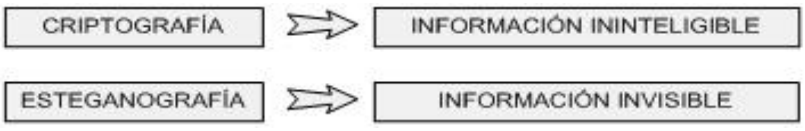

Figura 1: Diferencia entre Criptografía y Esteganografía [1]

También hay que mencionar que, muchos estudiosos incluyen la esteganografía como una rama que parte de la criptografía, Figura 2, siendo esta última la disciplina general.

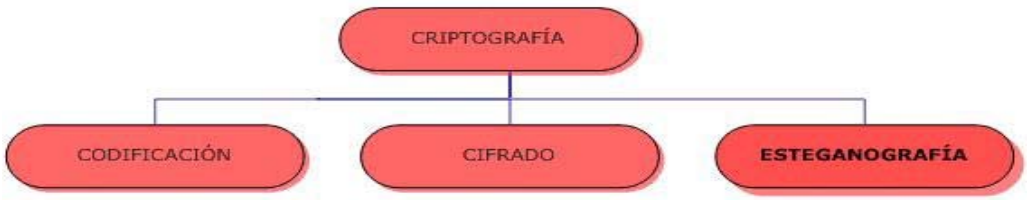

Figura 2: Esteganografía como rama de la Criptografía [1]

Muchos otros estudiosos, igualmente respetables, encuentran por contraposición, que la esteganografía es una disciplina separada de la criptografía, y por tanto no debería estar englobada bajo su ámbito. Incluso muchas personas califican a la esteganografía, y por ende al estegoanálisis, de arte. Debido a que históricamente la esteganografía se ha implementado de cierta forma creativa, usando medios portadores bastantes ingeniosos.

Debido al avance de la ciencia y las nuevas tecnologías los medios portadores usados actualmente están relacionados con el mundo digital de los ordenadores. Se pueden encontrar diversos soportes digitales para la ocultación de información en ellos.

- Archivos de imágenes, sonido, texto, video.
- Archivos ejecutables.
- Archivos de música y de películas.
- Páginas Web.
- Campos no usados de paquetes de redes (TCP/IP).
- Espacio no utilizado del disco: *slack space*.
- Particiones escondidas.
- HTML.

2.2 Esteganografía

A continuación se ilustra y esquematiza la síntesis de un proceso esteganográfico y su necesidad, teniendo en cuenta el conocido caso de "El problema del prisionero" (G. J. Simmons, 1983) [2].

"¿Cómo pueden comunicarse dos prisioneros, por ejemplo, para acordar un plan de fuga, si están en celdas separadas y todos los mensajes que intercambian pasan a través de un guardián?"[1]

Para solucionar este problema, del prisionero, no se debería usar sólo criptografía para establecer la comunicación de información sensible. Como ya se mencionó, cifrar los mensajes sólo oculta el significado pero no el hecho de que hay un mensaje. Si el guardián descubre que hay un mensaje cifrado, sospechará inmediatamente de los prisioneros y por lo tanto el guardián no permitirá que los mensajes lleguen a su destino. Además, los mensajes deben parecer inocuos, es decir, que no sean sospechosos de contener información sensible, como lo es el plan de fuga. Solamente así el guardián pasará el mensaje entre los prisioneros sin darse cuenta de que están tramando en su contra. Por último, el esquema debe estar preacordado entre las partes, para que las partes (los prisioneros) puedan descifrar o entender el mensaje.

La esteganografía ayudaría a resolver este problema. La Figura 3 ilustra el esquema típico de un estegosistema, sistema usado por la esteganografía:

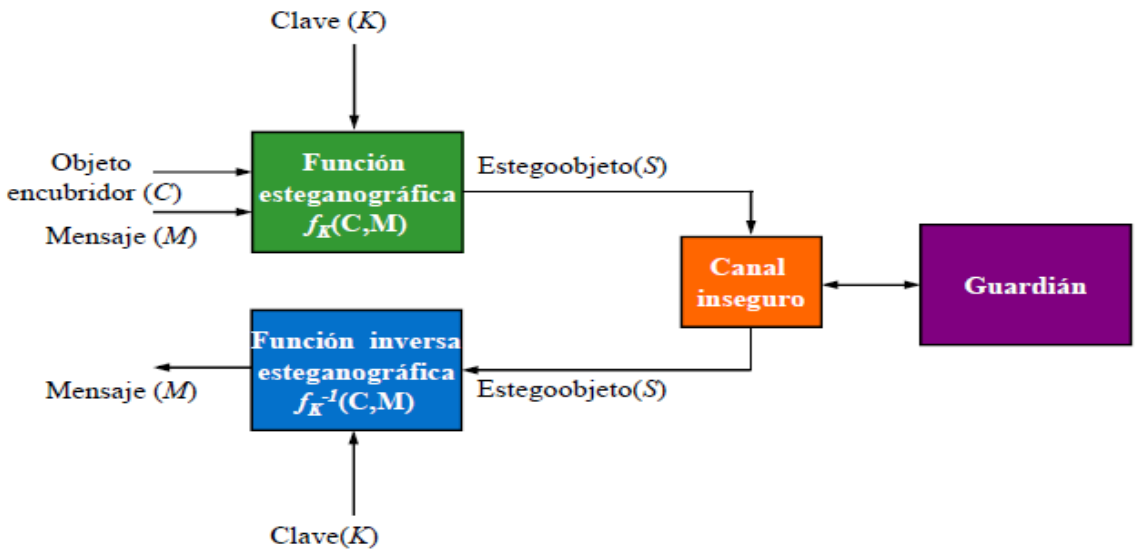

Figura 3: Diagrama de un estegosistema[2]

Inicialmente el *emisor* quiere enviar un mensaje *M*, por un *Canal Inseguro*, a un receptor. Dicho canal es inseguro, porque puede haber terceros escuchando por cierto canal, a estos terceros se les conoce por *Guardianes*.

[1] Cita de [3].

[2] Extraída de [3].

Para poder ocultar información el *emisor* usa un *Objeto Encubridor C*. Tanto *C* como *M* son la entrada de una *Función Esteganográfica,* la cual puede usar opcionalmente una Clave K para usarla en un algoritmo de cifrado y cifrar *M.* Si se cifra *M***,** el *emisor* tendrá que enviar al receptor la *Clave K* usada, por un *Canal Seguro*, que no sea escuchado por terceros. La salida de la función es un *Estego-objeto S*, de apariencia muy similar a *C*, pero conteniendo en su interior a *M*. Después el *emisor* envía por el *Canal Inseguro* el *Estego-objeto S*. Si el *Guardián* puede ver el envío de *S*, sólo verá, un objeto aparentemente igual al *Objeto Encubridor C*, lo que no le alertará de un posible envío de información confidencial de carácter sensible entre el *emisor* y el *receptor*. Una vez le ha llegado al *receptor S*, este deberá realizar la operación contraria que hizo el *emisor*, para poder extraer el *Mensaje M*, original. Esto lo hace con la *Función Inversa Esteganográfica*, que recibe como entrada *S*, y la *Clave K*, si el emisor cifró *M*, obteniendo este en última instancia.

La Función Esteganográfica, depende del mecanismo que se quiera usar, existiendo varios tipos de técnicas[3]:

- **Adición:** Se oculta el mensaje secreto en las secciones del medio portador que pueden ser ignoradas por la aplicación que lo procesa.
- **Generación:** Se crea el esteganograma a partir de la información secreta, sin contar con un medio portador previamente.
- **Substitución:** Se modifican ciertos datos del medio portador por los datos del mensaje secreto.
    - Principales dominios y métodos de la substitución:
        - **Dominio del Espacio: LSB (*Least-Significant Bit*)** o Bit menos significativo.
        - **Dominio de las Transformadas:** La transformación matemática de la información:
            - ***Discrete Cosine Transform* (DCT)** o Transformación discreta del Coseno.
            - ***Discrete Fourier Transform* (DFT)** o Transformación discreta de Fourier (DFT).
            - ***Discrete Wavelet Transform* (DWT)** o Transformación discreta Ondulada (tipo especial de transformada de Fourier).
    - Medios portadores (archivos digitales) más usados en substitución:
        - Archivos de música**.**
        - Archivos de imágenes.

[3] Tipos de técnicas extraídas de [4].

2.2.1 Esteganografía en Imágenes

Debido a que JUBSAC se centra en la esteganografía en imágenes, en las siguientes líneas se describen algunas las técnicas de substitución. Antes de ver en detalle las técnicas, se va a explicar, de forma muy resumida, cómo están construidas las imágenes, por si el lector no estuviese familiarizado con este concepto.

Las imágenes pueden tener varios modelos de color (RYB, RGB, CMYK,…), estos dan las reglas básicas para mezclar colores y conseguir el efecto deseado combinando estos. Uno de los modelos más utilizados es el RGB (*Red, Green, Blue* ó Rojo, Verde, Azul). El Modelo de Color RGB suma colores para obtener nuevos colores, como puede verse en la Figura 4:

Figura 4: Modelo RGB

Los colores se representan con 24 bits. 8 bits para cada componente RGB (3 componentes o Tres canales de color: Rojo (R), Verde (G), Azul (B)). Esto da 256 posibles niveles de color por cada canal. A continuación se van a explicar más en detalle los métodos de substitución:

- **Técnicas LSB:**

  Least-Significant Bit o Bit menos significativo [5] es una técnica que se basa en embeber los datos en los bits menos significativos de los píxeles de la imagen. Es decir, se descomponen el mensaje a ocultar en bits y estos se cambian únicamente por los bits menos significativos del archivo portador (imagen).

  Por ejemplo para ocultar el byte formado por los bits **11010001**, usando un bit por byte de la imagen, se cambiarán los bits originales por los que contienen la información oculta.

  Se tienen los siguientes 8 bytes, conjuntos de 8 bits, del archivo portador:

  00101100 00010110 01111000 00011110 10000001 00101101 01001100 01011000

  Si se oculta en el bit menos significativo, los bytes de la imagen quedan en:

  00101101 00010111 01111000 00011111 10000000 00101100 01001100 01011001

  Este reemplazo de bits no es inocuo, Figura 5, a cambio la imagen original se ve afectada en mayor o menor medida según se usen más bits de cada byte.

Figura 5: Variación de la calidad de la imagen usando LSB[4]

Para que el proceso de ocultación sea más seguro, y evitar el estegoanálisis, está técnica de ocultación se puede aleatorizar. Es decir, se ocultarán los datos en los bits menos significativos de los píxeles de la imagen en orden pseudo-aleaotrio y no siempre en los mismos bits. Incluso para hacer este proceso pseudo-aleatorio más aletorio y complejo se han llegado a utilizar redes neuronales [6] para elegir que bits usar para ocultar la información.

- **<u>Transformaciones Matemáticas:</u>**

  Con estas técnicas no se transforma directamente la imagen sino la información matemática que esta contiene al ser tratada como una señal.

  Las transformaciones más usadas son las tres mencionadas previamente, DCT, DFT y DWT. Estas usan integrales, pero para poder ser usadas desde un ordenador se aproximan de forma discreta usando sumadores. Estas funciones[5] son invertibles, es decir tiene una función inversa única.

  - **DCT[6]:**

$$f_j = \frac{1}{2}(x_0 + (-1)^j x_{n-1}) + \sum_{k=1}^{n-2} x_k \cos\left[\frac{\pi}{n-1} j\right]$$

[4] Figura de [4].

[6] Ecuación extraída de la Wikipedia, la enciclopedia libre.
<http://es.wikipedia.org/wiki/Transformada_de_coseno_discreta> [Octubre de 2010]

- **DFT[7]:**

$$f_j = \sum_{k=0}^{n-1} x_k e^{-\frac{2\pi i}{n} jk} \qquad j = 0, \ldots, n-1$$

- **DWT[8]:**

$$y[n] = (x * g)[n] = \sum_{k=-\infty}^{\infty} x[k] g[n-k].$$

El proceso de ocultación es relativamente sencillo, pero el algoritmo esteganográfico es más seguro ya que es más difícil de detectar este proceso.

Se describe, a continuación un ejemplo de algoritmo para la transformada del coseno (DCT):

1. Calcular la DCT de la imagen

2. Sustituir los coeficientes menores que un cierto valor umbral por bits de la información a ocultar

3. Calcular DCT-1 de la imagen

4. Almacenar

(La extracción es trivial aplicando el procedimiento inverso)

La transformación de una imagen a su DCT se puede ver en la Figura 6. Su inversa dará la imagen original.

Figura 6: Resultado de calcular la DCT a una imagen

Finalmente, destacar que, algunas imágenes son mucho mejores que otras cuando se utilizan como archivo portador. Por ejemplo, una imagen es más recomendable para la ocultación cuantos más pequeños detalles tenga.

[7] Ecuación extraída de la Wikipedia, la enciclopedia libre. <http://es.wikipedia.org/wiki/Transformada_de_Fourier_discreta> [Octubre de 2010]

[8] Ecuación extraída de la Wikipedia, la enciclopedia libre. <http://en.wikipedia.org/wiki/Discrete_wavelet_transform> [Octubre de 2010]

2.3 Estegoanálisis

El estegoanálisis, trata de romper las técnicas esteganográficas. El estegoanálisis tiene varios objetivos:

- **Detección:** Detecta contenido oculto en medios portadores, de varias formas:
  - Visual o Auditiva
  - Estructural
  - Estadística (Entropía, etc.)
- **Extracción:** Elimina la información oculta del medio portador.
- **Confusión:** Altera e introduce nueva información para dejar inservible la información oculta.
- **Deshabilitación:** Elimina la información oculta.

Si el estegoanálisis, consigue alguno de estos objetivos, se considera que la esteganografía ha fallado en su propósito. Por esto, con sólo con saber que un medio portador tiene información oculta el éxito es mayúsculo. Aún así, está claro, que lo óptimo e ideal será ser capaces de saber cuál es el mensaje oculto contenido en el medio portado y si estuviera cifrado, poder descifrarlo.

Destacar, que JUBSAC se centrará en el objetivo esteganográfico de la Detección, de información oculta en imágenes. El estegoanálisis de imágenes es un proceso complicado de hacer, porque hay tantos parámetros que se pueden tener en cuenta al hacer un algoritmo esteganográfico que estegoanalizar es bastante tedioso y complejo. Además, generalmente, un análisis no sirve para otro algoritmo, debido a que cada aplicación es distinta en funcionamiento y en origen.

Los formatos de imágenes más empleados por las herramientas esteganográficas y por los usuarios en general, son:

- **JPG ó JPEG:** *Joint Photographic Experts Group* (Grupo conjunto de expertos en fotografía).
- **BMP:** *Windows BitMaP (*Mapa de bits de ventanas*)*.
- **GIF:** *Graphics Image Format* (Formato de imágenes gráficas).
- **PNG**: *Portable Network Graphics* (Gráficos portables en red).

Y las técnicas usadas por los programas de esteganografía son las ya mencionadas, sobre el Dominio del Espacio y el Dominio de las Transformadas:

- **LSB:** Bit menos significativo.
- **DCT:** Transformada Discreta del Coseno.
- **DFT:** Transformada Discreta de Fourier.
- **DWT:** Transformada Discreta Ondulada.

El estegoanálisis se centra en atacar la esteganografía, habiendo varios métodos de ataque, según a que información se tenga acceso. Se pueden observar en la Tabla 1 un resumen:

El atacante emplea:	Cuando el atacante tiene acceso a:			
Ataque	Estego-imagen	Cubierta	Mensaje	Algoritmo
Estego-solo	X			
Cubierta conocida		X		
Mensaje conocido			X	
Estego elegido	X			X
Mensaje elegido			X	X

Tabla 1: Tipos de ataques del estegoanálisis[9]

Hay técnicas del estegoanálisis que se centran en romper ciertos algoritmos esteganográficos concretos (este estegoanálisis es conocido como *Targeted Steganalysis* o Estegoanálisis Objetivo), centrándose para ellos en las técnicas particulares usadas por estos para ocultar la información. Para ello disponen de qué algoritmo usaron y el estego-objeto (o estego-imagen) o el mensaje original. Pero si para el análisis sólo se dispone del estego-objeto, una imagen, y no se sabe qué herramienta se usó, ni siquiera si esa imagen tiene información oculta, entonces este tipo de análisis no sirve.

Existe, por contraposición, un tipo de análisis que es llamado Universal o a Ciegas (*Universal StegAnalysis, Blind StegAnalysis*) [8]. Este análisis pretende detectar la presencia de información oculta disponiendo únicamente de la información dada por el medio digital. Dicho análisis (estego-solo) es más complicado de realizar, pero también más potente, si este funciona correctamente, además es el que más tendrá lugar. Dichas técnicas usan análisis estadísticos, ya que las técnicas de LSB alteran la imagen original, pero el tipo de técnicas basadas en transformaciones matemáticas (DCT, DFT, DWT) hacen que el análisis estadístico, del mismo tipo, no sirva, aunque en cambio se pueden hacer análisis estadísticos diferentes.

El estegoanálisis es una técnica muy compleja, y por tanto, también lo es la creación de estegoanalizadores. La inteligencia artificial puede ayudar a conseguir estegoanalizadores de forma automática, que se tardaría mucho en crear de forma manual. Numerosos artículos [9] versan sobre como estegoanalizar medios digitales usando la inteligencia artificial, pero la mayoría de ellos no son reproducibles ni tangibles, ya que la información dispuesta en el artículo, suele ser, insuficiente y porque además, no existe una herramienta software para hacerlo. Los creadores hicieron ciertas tareas repetitivas a mano, o con programas ejecutados por lotes de los cuales no hay una versión disponible para su uso, y si se dispusieran de ellos no serían fácilmente entendibles ni ejecutables.

[9] Tabla extraída de [7].

2.4 Usos de la Esteganografía y el Estegoanálisis

El uso del estegoanálisis está muy vinculado con la seguridad en la información y con la seguridad nacional en el mundo real. Se conocen casos [10, 11] en los que terroristas han usado la esteganografía para intercambiar información. Tras los ataques del 11-S, las agencias de seguridad americanas, han dispuesto de más medios para detectar información oculta por grupos terroristas [12, 13]. También ha habido casos de esteganografía relacionados con intercambio de información de pedofilia [14]. Y un caso reciente, del mismo año 2010, donde espías del servicio de inteligencia de Rusia [15], usaban programas esteganográficos para ocultar mensajes de texto cifrados en imágenes, fue descubierto por el FBI. Otro caso conocido es el rumor [16] que cuenta que el departamento de justicia de los Estados Unidos, sacó de forma ilegal, datos financieros fuera de la agencia, ocultándolos en imágenes.

Se puede observar que la esteganografía puede ser usada para fines malignos y es muy complicado luchar contra ella. Esto es un motivo adicional para realizar nuestra aplicación. El uso de buenos sistemas estegoanalíticos puede ayudar a detener a terroristas, al espionaje industrial y a enemigos públicos de cualquier país que atentan contra la vida de sus habitantes. Para intentar también ayudar en el camino de luchar contra la esteganografía usada para fines malignos, se han realizado estudios que analizan imágenes en Internet [17]. En este estudio, los autores descargaron dos millones de imágenes desde las subastas de *eBay*, pero los resultados fueron negativos ya que no se encontró información oculta en ellas, por lo que no se pudo demostrar nada.

Para luchar contra un posible aumento de la esteganografía con malos usos, se ha pensado en tomar medidas para analizar los archivos que se mandan por servicios de mensajería, redes sociales y servicios de correos [18]. Finalmente, si se encontrara información oculta, esta sería alterada. Esta idea ha sido rechazada por su gran falta de privacidad con respecto a los usuarios legítimos.

Todos estos casos vistos, indican que el crear cada vez mejores estegoanalizadores, de forma dinámica, según se descubren nuevos algoritmos esteganográficos, es la solución para luchar contra el uso de la esteganografía en entornos en los que su uso pueda ser contraproducente.

2.5 Inteligencia Artificial y Aprendizaje Automático

La inteligencia artificial es la disciplina científico-técnica que trata de crear sistemas artificiales capaces de comportamientos que, imiten la inteligencia humana[10]. El aprendizaje automático, es una rama de la inteligencia artificial cuyo objetivo es desarrollar técnicas que permitan a las computadoras aprender. Los algoritmos de aprendizaje automático intentan crear programas capaces de generalizar comportamientos a partir de una información no estructurada suministrada en forma de ejemplos[11]. Es, por lo tanto, un proceso de inducción del conocimiento. Existen dos grupos principales:

- **Aprendizaje Supervisado**: Aplica procesos inductivos a ejemplos ya clasificados.
    - **Clasificación**: Las clasificaciones (o clases) de los ejemplos son valores discretos.
    - **Regresión**: Las clasificaciones de los ejemplos son valores continuos.

- **Aprendizaje No Supervisado**: Todo el proceso de modelado se lleva a cabo sobre un conjunto de ejemplos formado tan sólo por entradas al sistema. No se tiene información sobre las categorías de esos ejemplos.
    - **Agrupación** (o ***clustering***): Se basa en detectar agrupaciones naturales en los datos.

Como técnicas de clasificación destacan:

- **Las redes de neuronas artificiales**: Son un paradigma de aprendizaje automático, inspirado en la forma en que funciona el sistema nervioso de los animales. Se trata de un sistema de interconexión de neuronas en una red que colabora para producir un estímulo de salida[12].
- **Los árboles de decisión:** Son unos modelos de predicción, donde dada una base de datos, se construyen diagramas de construcciones lógicas, muy similares a los sistemas de predicción basados en reglas, que sirven para representar y categorizar una serie de condiciones que ocurren de forma sucesiva, para la resolución de un problema[12].
- **Aprendizaje Perezoso (*Lazy Learning*)**: Sirven para estimar la probabilidad a posteriori de que un elemento pertenezca a una clase a partir de la información proporcionada por el conjunto de ejemplos. Este método supone que los ejemplos (o vecinos) más cercanos dan la mejor clasificación[12].
- **Las redes bayesianas:** Son unos modelos probabilísticos multivariados que relacionan un conjunto de variables aleatorias mediante un grafo dirigido que indica explícitamente influencia causal. Gracias a su motor de actualización de probabilidades, el Teorema de Bayes, las redes bayesianas son una herramienta extremadamente útil en la estimación de probabilidades ante nuevas evidencias[12].

[10] Definición de [19]

[11] Definición de [20]

[12] Definición de [21]

3. ANÁLISIS

En este capítulo se abordará el estado del arte, para una labor de documentación y análisis sobre la literatura existentes. Posteriormente se tratará el análisis para la extracción de funcionalidades del marco a desarrollar, donde se desarrollarán los casos de uso.

3.1 Estado del Arte

El estado del arte aborda las técnicas esteganográficas y el estegoanálisis.

3.1.1 Técnicas Esteganográficas

A continuación, se estudiará, desde lo más general a lo más específico la esteganografía y sus herramientas de ocultación, profundizando en la ocultación de información en imágenes.

En la Figura 7 se puede ver que el número de artículos sobre esteganografía y marcas de agua por el IEEE [22] ha aumentado notablemente en los últimos años. Son temas en auge, y se prevé que seguirá en expansión. De hecho desde el 2007 hasta el día de hoy se han publicado unos 6630 artículos que tratan la esteganografía [23]. Es decir, en los últimos 4 años se han publicado unos 1600 artículos por año.

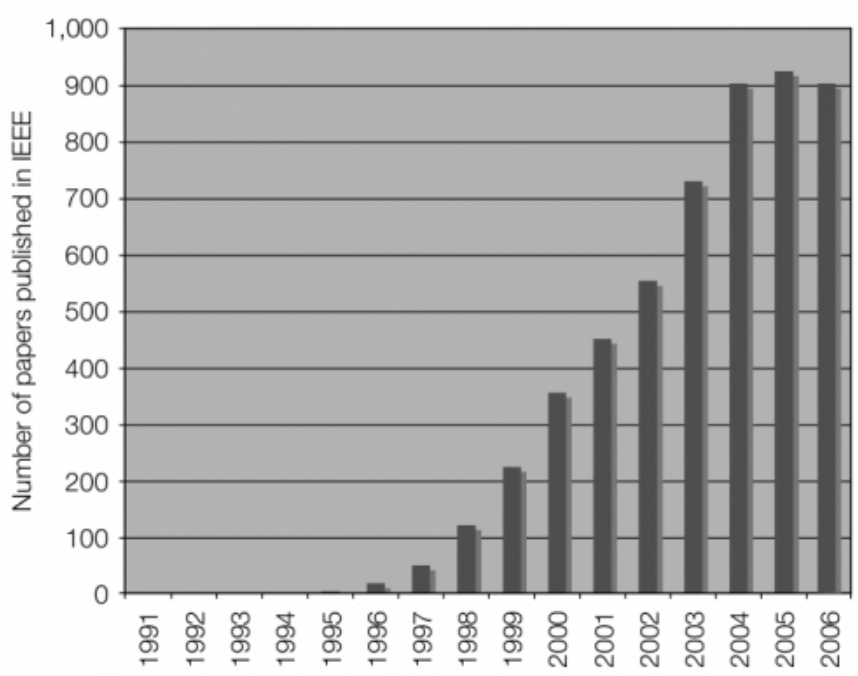

Figura 7: Número de artículos sobre esteganografía en los últimos años

A continuación, se explicarán en detalle algunas de las herramientas esteganográficas más utilizadas, centrándose en los formatos de imágenes con los que trabajan:

3.1.1.1 OpenStego

OpenStego v0.5.2 [24] es una herramienta desarrollada en Java para la esteganografía, bajo licencia *GNU General Public License* 2.0 (GPL) [25], la cual permite la copia y modificación del software, pero no la modificación de la licencia. Soporta cifrado opcional de los datos, antes de embeber en la imagen. Se pueden añadir complementos (*plugins*) para diferentes algoritmos esteganográficos. Los algoritmos que incluye en esta versión son:

1. **LSB**: Embebiendo los datos en los bits menos significativos de los píxeles de la imagen.
2. **RandomLSB**: Oculta los datos en los bits menos significativos de los píxeles de la imagen en orden pseudo-aleaotrio, para evitar el estegoanálisis.

La herramienta dispone de ejecución por interfaz gráfica de usuario (GUI) y por línea de comandos. Admite como entrada los formatos: BMP, JPG ó JPEG, GIF, PNG y WBMP, pero como salida sólo tiene el formato PNG. Además comprime los datos a ocultar y también los puede cifrar usando una contraseña.

3.1.1.2 Hide4PGP

Hide4PGP v2.0 [26] es un programa que se distribuye gratuitamente, disponible para su uso y por tiempo ilimitado (llamado a este tipo de software como *freeware*). Disponible el código fuente en ANSI C y ejecutables precompilados para Linux, DOS, OS/2, y consola de Win32 (9x y NT). Esta herramienta tiene como propósito ocultar cualquier dato en BMP, WAV o VOC, siendo su salida el mismo formato de entrada.

Hide4PGP fue diseñado, como un complemento a las versiones 2.x del poderoso programa libre de cifrado *Pretty Good Privacy* (PGP) [27] creado por Phil Zimmermann y para Stealth [28], creado por Henry Hastur, un programa que usa un mensaje cifrado con PGP 2.x y elimina cualquier tipo de cabeceras para garantizar que el resultado se parece a ruido aleatorio.

Como características importantes de la versión 2.0 se tienen:

- Almacena y recupera la longitud exacta de los datos ocultos.
- Distribuye los datos ocultos uniformemente sobre el archivo multimedia.
- El formato de salida es robusto contra conversiones del formato. Sólo los formatos de compresión con pérdida de calidad, provocarán la pérdida también de los datos ocultos

Hide4PGP permite que el archivo de datos pueda ser convertido a otros formatos de imagen o sonido y dejar los datos intactos. Se propaga la información secreta de forma uniforme sobre todo el archivo multimedia (cubierta). Es capaz de utilizar más de un bit por pixel. Calcula los límites de la cantidad de bits que puede usar: con archivos BMP de paletas SVGA, 1 bit, con archivos BMP en escala de grises, 2 bits, y con archivos BMP de color verdadero, 6 bits por píxel (2 bits por canal). Los límites deben garantizar que ningún cambio en la cubierta será evidente visualmente.

3.1.1.3 GifShuffle

GifShuffle v2.0 [29] es un programa de dominio público y el código fuente es en ANSI C y está abierto. Oculta mensajes en imágenes de formato GIF, barajando el mapa de colores. Una imagen barajada es visualmente indistinguible de la original.

GifShuffle trabaja con todas las imágenes GIF, incluyendo aquellos con transparencia y animación. Además proporciona compresión y cifrado para el mensaje a ocultar. El esquema de compresión usado por GifShuffle es un esquema bastante rudimentario de la Codificación de Huffman [30], donde las tablas se optimizaron para texto inglés. Dependiendo del texto del mensaje, se puede entre un 25 y un 40% de compresión. El algoritmo de cifrado es ICE [31], un cifrador de bloque de 64 bits, diseñado por el propio autor de GifShuffle. Este ejecuta en modo CFB. El modo *cipher feedback* (CFB), hace que el cifrado en bloque opere como una unidad de flujo de cifrado.

El funcionamiento de GifShuffle se basa en que cualquier lista de n elementos se pueden ordenar n! maneras, lo que significa que cualquier orden en particular se puede representar como un número dentro del rango [0, n! -1]. Este número a su vez puede almacenar aproximadamente log2 (n!) bits de información. Por lo tanto, en una imagen GIF con 256 colores se pueden almacenar hasta 1675 bits (209 bytes) de información por el intercambio de los colores en su mapa de colores.

3.1.1.4 F5

F5 v0.1 [32] es un paquete implementado en Java, destinado a mostrar un nuevo algoritmo esteganográfico explicado en el artículo [33] *"High Capacity Despite Better Steganalysis (F5–A Steganographic Algorithm)"* por el propio autor del programa. Es una versión preliminar para embeber archivos en imágenes de color verdadero BMP, GIF y JPG o JPEG, siendo la salida JPG o JPEG. Permite usar contraseñas para dar seguridad y permite elegir la calidad que tendrá la imagen final, con respecto a la original, para comprimir esta según se elija.

F5 soporta los ataques visuales y estadísticos, además ofrece una capacidad esteganográfica mayor. F5 implementa codificación por matrices para mejorar la eficiencia de incrustación. Por lo tanto, reduce el número de cambios necesarios. F5 emplea permutaciones para difundir de forma uniforme los cambios sobre los esteganogramas (el resultado de la ocultación). Posteriormente a su aparición se han escrito artículos sobre cómo romper este algoritmo [34].

3.1.1.5 Steghide

Steghide v0.5.1 [35] es un programa de estenografía, bajo licencia GNU *General Public License* 2.0 (GPL) [25], que permite ocultar datos en varios tipos de imágenes y archivos de audio. Los respectivos muestreos de frecuencia de color no varían, lo que hace que el estego-objeto soporte pruebas estadísticas de primer orden. Funciona por línea de comandos.

Sus características incluyen la compresión y el cifrado de los datos embebidos, y revisión automática de la integridad usando un *checksum* (suma de verificación, es una forma de

control de redundancia). El *checksum* se calcula usando el algoritmo CRC32. Se reconocen los formatos de archivo JPEG, BMP, WAV y AU para usar como cubiertas, y sus salidas tienen el mismo formato que la entrada. No existen restricciones en el formato de los datos a ocultar.

El algoritmo de ocultación primero comprime y cifra la información secreta. Este intercambia valores de píxeles, por lo que las estadísticas de primer orden (i.e. las veces que se usa un color en la imagen) no se alteran. El algoritmo de cifrado, por defecto, es el Rijndael [36], con un tamaño de clave 128 bits (el cual es AES (*Advanced Encryption Standard*)), en modo CBC. *Advanced Encryption Standard* (AES), también conocido como Rijndael, es un esquema de cifrado por bloques adoptado como un estándar de cifrado por el gobierno de los Estados Unidos. En el modo CBC (*cipher block chaining*), antes de ser cifrado, a cada bloque de texto se le aplica una operación XOR con el previo bloque ya cifrado. De este modo, cada bloque es dependiente de todos los bloques de texto planos hasta ese punto. Además, para hacer cada mensaje único se puede usar un vector de inicialización.

Además, se pueden usar otros algoritmos en distintos modos de operación:

- **Cast-128:** cbc cfb ctr ecb ncfb nofb ofb
- **Gost**: cbc cfb ctr ecb ncfb nofb ofb
- **Rijndael-128**: cbc cfb ctr ecb ncfb nofb ofb
- **Twofish**: cbc cfb ctr ecb ncfb nofb ofb
- **Arcfour**: stream (flujo)
- **Cast-256**: cbc cfb ctr ecb ncfb nofb ofb
- **Loki97**: cbc cfb ctr ecb ncfb nofb ofb
- **Rijndael-192**: cbc cfb ctr ecb ncfb nofb ofb
- **Saferplus**: cbc cfb ctr ecb ncfb nofb ofb
- **Wake**: stream (flujo)
- **DES**: cbc cfb ctr ecb ncfb nofb ofb
- **Rijndael-256:** cbc cfb ctr ecb ncfb nofb ofb
- **Serpent**: cbc cfb ctr ecb ncfb nofb ofb
- **Xtea**: cbc cfb ctr ecb ncfb nofb ofb
- **Blowfish**: cbc cfb ctr ecb ncfb nofb ofb
- **Enigma**: stream (flujo)
- **RC2**: cbc cfb ctr ecb ncfb nofb ofb

- **Triple DES**: cbc cfb ctr ecb ncfb nofb ofb

3.1.1.6 Blindside

Blindside v0.9 [37] es un programa de estenografía, de código abierto y gratuito, para imágenes BMP, y la salida generada también es BMP. Escrito en ANSI C. El software crea modificaciones ligeras de color en una imagen que, aunque invisible al ojo humano, puede crear un una gran cantidad de espacio para contener información. Algunas utilidades esteganográficas modulan y cambian cada píxel en la imagen, y aunque esto puede crear grandes espacios de almacenamiento, a menudo los efectos son notables en la imagen. Por el contrario Blindside calcula las diferencias de color entre píxeles y sólo se modificará la imagen donde los cambios se consideran insignificantes perceptivamente. Un algoritmo base de cifrado XOR también se puede utilizar para codificar los datos con una contraseña secreta. Blindside se ejecuta por línea de comandos sobre muchas plataformas:

- Win 95/98/NT/2000 Intel.
- Solaris en SPARC.
- IBM AIX en AS6000.
- HPUX 11.
- Linux en Intel x86.

3.1.1.7 Vecna

Vecna v0.6 [38] es una herramienta que oculta un mensaje secreto dentro de una imagen. Después de este proceso de ocultación la imagen resulta idéntica a la original, a simple vista. Vecna, está escrito completamente en Java, el cual hace esta plataforma independiente. Se encuentra bajo licencia GPL [25]. Almacena la información del mensaje secreto y los bits menos significativos de cada canal de un píxel (LSB). Como características principales se encuentran las siguientes:

- Oculta cualquier tipo de datos dentro de una imagen.
- Muchos tipos de imágenes soportados (BMP, JPG, PNG, GIF,...), pero la salida siempre será de PNG de 32 bpp (bits per pixel).
- Barajeo aleatorio de los datos para dar más seguridad a la ocultación.
- Sólo la contraseña deberá ser compartida por un canal seguro.
- Se pueden ocultar mensajes cifrados, pero la herramienta no posee algoritmos de cifrado.
- Ejecución por Interfaz Gráfica y por Línea de Comandos.

3.1.1.8 JPHide

JPHide v0.5 [39] es un programa que permite ocultar un archivo en una imagen JPEG, siendo por tanto la entrada y la salida una imagen JPEG o JPG. El objetivo del programa no es simplemente ocultar un archivo, sino hacerlo de tal manera que sea imposible probar que el archivo portador contiene un archivo oculto.

La herramienta viene acompañada de otra para el proceso de extracción de la información oculta. Se llama JPSeek. El código, en ANSI C, de las dos utilidades está disponible. La técnica usada para ocultar la información es LSB. Disponible versiones para Linux, v0.3, con línea de comandos, para Windows, v0.5, con interfaz gráfica (Jphswin) y también una versión ejecutable por línea de comandos para DOS.

3.1.1.9 JSteg

JSteg [40] es otra herramienta para ocultar información en imágenes JPEG o JPG. Usa el programa jpeg-4, para compresión de imágenes en JPEG o JPG, escrito en ANSI C, para modificar este y ampliarlo para que permita la ocultación de información. Se ejecutar por línea de comandos. Funciona en Linux, pero también existen algunas versiones compiladas para Windows.

JSteg cambia los coeficientes de la transformada discreta del coseno de la imagen original. El procedimiento de codificación JPEG divide una imagen en bloques de 8x8 píxeles en el espacio de color YCbCr. A continuación, estos bloques se ejecutan a través de una transformación discreta del coseno (DCT) y los coeficientes de frecuencia resultante se escalan para eliminar las que un espectador humano podría detectar en condiciones normales. Los bits de menor orden, o menos significativos, de todos los coeficientes de frecuencia, que no sean cero, se sustituyen por los bits sucesivos del archivo a ocultar. Y estos coeficientes modificados se envían a un codificador Huffmann.

3.1.1.10 OutGuess

OutGuess v0.2 [41] es una herramienta esteganográfica que permite la inserción de información oculta en los bits redundantes de fuentes de datos. La naturaleza de los datos es irrelevante para el núcleo de OutGuess. El programa se basa en datos de controladores específicos que van a extraer los bits redundantes y escribirán en ellos para modificarlos. Soporta los formatos de imágenes JPEG o JPG y PNM.

Es totalmente libre incluso para uso comercial ya que está bajo licencia software BSD (*Berkeley Software Distribution*) [42]. Esta licencia permite el uso del código fuente en software no libre. Usa el programa jpeg-6b, para compresión de imágenes en JPEG o JPG, escrito en ANSI C, para modificar este y ampliarlo para que permita la ocultación de información. Se ejecuta por línea de comandos. Funciona en Linux, pero también existen algunas versiones compiladas para Windows y el código está disponible.

OutGuess cambia los coeficientes de la transformada discreta del coseno de la imagen original. Para las imágenes JPEG, OutGuess, preserva las estadísticas basadas en el cálculo de frecuencias. Por lo tanto, los test estadísticos no pueden detectar la presencia de contenido esteganográfico. Antes de embeber datos en una imagen, OutGuess puede determinar el tamaño máximo del mensaje que puede ser ocultado para preservar las estadísticas basadas en el cálculo de frecuencias. OutGuess intenta encontrar una secuencia de bits que minimice el número de cambios en los datos.

3.1.1.11 Steganos

Steganos v1.4 [43] combina dos tecnologías para la seguridad de la información: criptografía y esteganografía, lo que es muy común en programas esteganográficos. Steganos oculta un archivo dentro de un BMP, VOC, WAV o fichero ASCII, no haciendo estos archivos inservibles para terceros (para esto hay que usar métodos criptográficos), pero si hace que

estos oculten la existencia de cualquier información sensible. Steganos funciona para Linux, Windows, MS-DOS y OS/2, usando línea de comandos.

Almacena información en el bit menos significativo, empezando después de la cabecera del archivo. Steganos no sólo se limita a almacenar el archivo, el tamaño del archivo y una suma de comprobación (*checksum*), sino que gracias a una contraseña, toda la información almacenada en el archivo se cifra. También crea ruido por lo que es más difícil detectar los bits que se han modificado, por ejemplo con rotación de la paleta de colores.

3.1.1.12 Stegtools

Stegtools v0.4c [44] es un programa el cual permite ocultar un archivo en una imagen BMP (soporta bitmap de 24bpp), siendo por tanto la entrada y la salida una imagen BMP. La herramienta está formada por dos ejecutables, stegwrite y stegread. Una oculta información y otra extrae esa información. El código fuente de ambas herramientas está disponible en ANSI C bajo licencia GPL [25]. La técnica de ocultación de información es LSB, donde oculta el mensaje en los bits menos significativos, los cuales pueden ser indicados por parámetro. La versión disponible de Stegtools es para plataformas UNIX, siendo ejecutada únicamente por línea de comandos.

3.1.1.13 DIIT

Digital Invisible Ink Toolkit (DIIT) [45] es una herramienta esteganográfica escrita en Java que puede ocultar cualquier conjunto de archivos dentro de una imagen digital (si el mensaje cabe y la imagen es de 24 bits de color (BMP, JPG, PNG)). La salida será una imagen en el mismo formato que el original. Esta funciona en Java, pero sólo dispone de GUI y no de ejecución por línea de comandos. Es software de código abierto bajo licencia GPL [25].

Además la herramienta dispone de un apartado para el estegoanálisis, con lo que se puede saber si las imágenes tienen información oculta o incluso se pueden realizar análisis estadísticos sobre ellas. En cuanto a la esteganografía usa una contraseña y dispone de varios algoritmos de ocultación para seleccionar:

- **BattleSteg:** Filtra la imagen para obtener una lista de los mejores lugares para ocultar. Después aleatoriamente va seleccionando puntos sobre la imagen hasta que un lugar, de los mejores, es encontrado. Durante un instante después la selección de puntos se agrupa alrededor del lugar, de los mejores, encontrado. Y a continuación se aleja y vuelve a seleccionar puntos aleatoriamente. Es el mejor de todos los algoritmos y además seguro porque usa una contraseña para recuperar la imagen.

- **BlindHide:** Empieza en el píxel (0,0) y se mueve a lo largo de cada pixel. Usa esteganografía pura.

- **Dynamic BattleSteg**: Funciona con el mismo mecanismo que BattleSteg, sólo que mejorado, ya que previene fugas de memoria, pero no es compatible con BattleSteg.

- **Dynamic FilterFirst:** Funciona con el mismo mecanismo que FilterFirst, sólo que previene fugas de memoria, pero no es compatible con FilterFirst.

- **FilterFirst:** Un método esteganográfico puro. Usa filtros de detección de bordes para obtener una lista ordenada de los mejores lugares para ocultar. Después escribe la

lista en el orden dado. Es seguro ya que usa una contraseña para recuperar la imagen original.

- **HideSeek:** Distribuye aleatoriamente el mensaje a lo largo de la imagen. Usa una contraseña para generar una semilla aleatoria. Funciona como BattleSteg, pero no es tan eficaz debido a que no oculta en los mejores lugares de la imagen.

Para estos algoritmos, la opción más importante es el número de bits a escribir (más bits cuanta más información se puede ocultar en una imagen). Sin embargo, cuanto más se escribe, menos oculta será. Los bits se numeran del LSB (bit menos significativo) al MSB (bit más significativo).

3.1.1.14 Hide & Reveal

Hide & Reveal v1.7.0 [46] es un software de esteganografía de código abierto y una librería en Java que se distribuye bajo la GNU GPL [25]. Está diseñado principalmente para los científicos que deseen experimentar técnicas nuevas de ocultación o de estegoanálisis en diversas cubiertas. Dispone de ejecución sólo por GUI y no por línea de comandos. Al estar en Java puede ser ejecutado en cualquier plataforma. Se tienen las siguientes características principales:

- Ejecución en varios hilos (Multi-threaded)
- Permite ocultar cualquier tipo de archivo en imágenes BMP, PNG y TIF.
- Usa todos los algoritmos de la librería org.steganography (usando las técnicas LSB).
- Personalizable ya que se pueden añadir como complementos nuevos algoritmos y nuevos formatos de cubiertas, cargados de forma dinámica en tiempo de ejecución, usando un archivo de configuración *XML*.
- Traducido en inglés y francés.

3.1.1.15 Hide and Seek

Hide and Seek v4.1 [47] es un programa de ocultación de información en imágenes GIF, de código abierto, escrito en ANSI C, pero sólo ha sido compilado para usar bajo DOS y Windows. Se ejecuta por línea de comandos, pero hay una nueva versión v5.0, la cual dispone de una GUI sencilla bajo DOS. Después hay una versión más moderna, Hide and Seek 95 v 1.1, la cual dispone de una mejor GUI y funciona para imágenes BMP. La imagen GIF debe ser de 256 colores (o en escala de grises) para que Hide and Seek funcione bien, sino la degradación provocada por la ocultación será evidente. Usa la técnica de **LSB** para la ocultación.

Puede cifrar, sólo la cabecera del archivo de información, usando una clave de al menos 8 caracteres, usando el cifrador IDEA [48]. IDEA, *International Data Encryption Algorithm* (Algoritmo Internacional de Cifrado de Datos) es un cifrador por bloques diseñado por Xuejia Lai y James L. Massey de la Escuela Politécnica Federal de Zúrich y descrito por primera vez en 1991. Fue un algoritmo propuesto como reemplazo del DES (*Data Encryption Standard*). A continuación, se muestra en una tabla resumen, Tabla 2, las características principales de las herramientas, donde las técnicas de ocultación son dos, como se trató previamente en los conceptos básicos (capítulo 2):

- **TDS:** *Transform Domain Steganography* (Dominio de las Transformadas).
- **SDS**: *Spatial Domain Steganography* (*LSB Replacement and LSB Matching*) (Dominio Espacial, Técnicas del Bit Menos Significativo).

Herramienta	Formatos Entradas	Formatos Salidas	Técnica Ocultación	Compresión	Clave	GUI	Línea de Comandos	Código disponible	Sistema Operativo
Blindside	BMP	BMP	SDS	No	Sí	No	Sí	Sí, ANSI C	Linux, Win, DOS
DIIT	BMP, JPG, PNG	BMP, JPG, PNG	SDS	No	Sí	Sí	No	Sí, Java	Todos
F5	BMP, JPEG ó JPG, GIF	JPEG ó JPG	TDS	No	Sí	No	Sí	Sí, Java	Todos
GifShuffle	GIF	GIF	Cambia el orden del mapa de colores	Sí	Sí	No	Sí	Sí, ANSI C	Linux
Hide and Seek	GIF	GIF	SDS	No	Sí	Sí	Sí	Sí, ANSI C	Win, DOS
Hide&Reveal	BMP, PNG	BMP, PNG	SDS	No	No	Sí	No	Sí, Java	Todos
Hide4PGP	BMP	BMP	SDS	No	No	No	Sí	Sí, ANSI C	Linux, Win, DOS, OS/2
JPHide	JPEG ó JPG	JPEG o JPG	SDS	No	No	Sí	Sí	Sí, ANSI C	Linux, Win, DOS
JSteg	JPEG ó JPG	JPEG ó JPG	TDS	No	No	No	Sí	Sí, ANSI C	Linux, Win, DOS
OpenStego	BMP, JPEG ó JPG, GIF, PNG	PNG	SDS	Sí	Sí	Sí	Sí	Sí, Java	Todos
OutGuess	JPEG ó JPG	JPEG ó JPG	TDS	No	Sí	No	Sí	Sí, ANSI C	Linux, Win, DOS
Steganos	BMP	BMP	SDS	No	Sí	No	Sí	No	Linux, Win, DOS, OS/2
Steghide	BMP, JPEG ó JPG	BMP, JPEG ó JPG	SDS	Sí	Sí	No	Sí	Sí, ANSI C	Linux, Win
Stegtools	BMP	BMP	SDS	No	No	No	Sí	Sí, ANSI C	Linux
Vecna	BMP, JPEG ó JPG, GIF, PNG	PNG	SDS	No	Sí	Sí	Sí	Sí, Java	Todos

Tabla 2: Listado de herramientas esteganográficas (para imágenes)

3.1.2 Estegoanálisis

A continuación, se estudiará el estegoanálisis en imágenes y el uso de la inteligencia artificial para construir estegoanalizadores, y como se ha usado esta para JUBSAC.

3.1.2.1 Inteligencia Artificial en el Estegoanálisis

Los algoritmos de la inteligencia artificial realizan su entrenamiento en base a una serie de datos para poder ofrecer resultados. Por lo que el problema principal es la recopilación de datos de imágenes. Existe una forma de abordar el problema, en la detección de información oculta en imágenes, que es usar métodos estadísticos para extraer información de las imágenes. A esta técnica se la conoce como ANOVA (*analysis of variance*, análisis de varianza). Las medidas estadísticas de ANOVA son llamadas IQM (*Image Quality Metrics*) [49, 50, 51, 52]. Se basan en listas de fórmulas que calculan la diferencia entre una imagen original y otra con información oculta, de formas diferentes. Un ejemplo de estas listas sería la mostrada en la Figura 8:

IMAGE QUALITY MEASURES

Average Difference	$AD = \sum_{j=1}^{M}\sum_{k=1}^{N}[F(j,k) - \hat{F}(j,k)] / MN$
Structural Content	$SC = \sum_{j-1}^{M}\sum_{k=1}^{N}[F(j,k)]^2 / \sum_{j=1}^{M}\sum_{k=1}^{N}[\hat{F}(j,k)]^2$
N. Cross-Correlation	$NK = \sum_{j=1}^{M}\sum_{k=1}^{N}F(j,k)\hat{F}(j,k) / \sum_{j=1}^{M}\sum_{k=1}^{N}[F(j,k)]^2$
Correlation Quality	$CQ = \sum_{j=1}^{M}\sum_{k=1}^{N}F(j,k)\hat{F}(j,k) / \sum_{j=1}^{M}\sum_{k=1}^{N}F(j,k)$
Maximum Difference	$MD = \text{Max}\{\lvert F(j,k) - \hat{F}(j,k)\rvert\}$
Image Fidelity	$IF = 1 - (\sum_{j=1}^{M}\sum_{k=1}^{N}[F(j,k) - \hat{F}(j,k)]^2 / \sum_{j=1}^{M}\sum_{k=1}^{N}[F(j,k)]^2)$
Weighted Distance	WD: Every element of the difference matrix is normalized in some way and L_1-norm is applied [1].
Laplacian Mean Square Error	$LMSE = \sum_{j=1}^{M-1}\sum_{k=2}^{N-1}[O\{F(j,k)\} - O\{\hat{F}(j,k)\}]^2 / \sum_{j=1}^{M-1}\sum_{k=2}^{N-1}[O\{F(j,k)\}]^2$
Peak Mean Square Error	$PMSE = \frac{1}{MN}\sum_{j=1}^{M}\sum_{k=1}^{N}[F(j,k)\} - \hat{F}(j,k)]^2 / [\text{Max}\{F(j,k)\}]^2$
N. Absolute Error	$NAE = \sum_{j=1}^{M}\sum_{k=1}^{N}\lvert O\{F(j,k)\} - O\{\hat{F}(j,k)\}\rvert / \sum_{j=1}^{M}\sum_{k=1}^{N}\lvert O\{F(j,k)\}\rvert$
N. Mean Square Error	$NMSE = \sum_{j=1}^{M}\sum_{k=1}^{N}[O\{F(j,k)\} - O\{\hat{F}(j,k)\}]^2 / \sum_{j=1}^{M}\sum_{k=1}^{N}[O\{F(j,k)\}]^2$
L_p-norm	$L_p = \{\frac{1}{MN}\sum_{j=1}^{M}\sum_{k=1}^{N}\lvert F(j,k) - \hat{F}(j,k)\rvert^p\}^{1/p}, p = 1,2,3$
Hosaka plot	A graphical quality measure. The area and shape of the plot gives information about the type and amount of degradation [1,6].
Histogram	Another graphical quality measure. Gives the probability distribution of the pixel values in the difference image.

Note: For LMSE, $O\{F(j,k)\}=F(j+1,k)+F(j-1,k)+F(j,k+1)+F(j,k-1)-4F(j,k)$. For NAE, NMSE, and L_2-norm, $O\{F(j,k)\}$ is defined in three ways: (1) $O\{F(j,k)\}=F(j,k)$, (2) $O\{F(j,k)\}=F(j,k)^{1/3}$, (3) $O\{F(u,v)\}=H\{(u^2+v^2)^{1/2}\}F(u,v)$ (in cosine transform domain).

Figura 8: Image Quality Measures (IQMs) [49]

Después se usan, las diferencias, como datos que forman las instancias para entrenar algoritmos de aprendizaje automático. Para ello se usa una muestra grande de imágenes y una serie de algoritmos esteganográficos.

En estudios como [53, 54], por ejemplo, se entrenan redes neuronales. El proceso seguido se basa en la división de cada imagen de entrenamiento en bloques de 8x8. A continuación, se calculan las transformadas DCT y DFT de cada bloque y se extraen las IQM, comparando las imágenes sin y con información oculta. También, se calcula la transformada DWT, con tres niveles de profundidad, de cada imagen y finalmente se calcula la media, la varianza, la desviación típica y la curtosis, de cada nivel. Con toda esta información se entrena a una red de neuronas.

En otros estudios publicados como [55, 56] se emplean sistemas híbridos para solucionar el problema de detectar información oculta en imágenes usando IQM. En la Tabla 3 se puede ver el proceso de construcción del estegoanalizador en los estudios mencionados[13]:

Fase: *Aprendizaje* ***Entrada:*** *Una base de datos de imágenes* ***Salida:*** *Una base de conocimiento capaz de discriminar entre una imagen limpia o con información oculta.*
1. Construcción de la base de datos de imágenes: Se prepara una base de datos de imágenes con imágenes limpias y con imágenes con información oculta, generadas a partir de diferentes sistemas esteganográficos. **2. Eliminación de la dependencia del contenido:** Una única señal aleatoria de referencia que es común a todas las señales es seleccionada para la evaluación de IQM. **3. Evaluación por IQM:** Varios IQM como medidas de la diferencia entre píxeles, medidas basadas en la correlación, medidas basadas en bordes, medidas basadas en la distancia espectral, medidas basadas en el contexto, son evaluadas entre la señal de test y la señal de referencia. **4. Selección de Características basado en un Algoritmo Genético:** Las características independientes del contenido, que son sensibles a los datos ocultados, son seleccionados basándose en una estrategia de búsqueda genética. **5. Creación de un conjunto de datos**: Se crea un conjunto de datos a partir de las características seleccionadas. **6. Entrenamiento:** El estegoanalizador se somete al aprendizaje aplicando el algoritmo X-Medias sobre el conjunto de datos y se construye una base de conocimiento. **7. Sistema Preparado:** El estegoanalizador ya está listo para estegoanalizar de forma universal y a ciegas.

Tabla 3: Creación de un estegoanalizador usando IQM

Se crea un estegoanalizador usando Algoritmos Genéticos y X-Medias, que luego se ejecutará como un proceso "demonio", que analiza el contenido multimedia de una red para detectar si se está usando esteganografía.

El estegoanalizador creado con IQM necesita como entrada dos imágenes: con información oculta (o sospechas de que contiene información oculta) y sin ella, por lo que no es factible para su implantación real y sólo sirve para ataques en los que se cuente con la imagen cubierta y el estagoobjeto.

[13] Proceso extraído de [55] y [56].

Además, el algoritmo utilizado es X-Medias, el cual funciona agrupando en conjuntos (*clusters*) las instancias con las que aprende, sin tener una clase que defina a cada instancia. Simplemente separará en conjuntos los ejemplos que compartan características comunes.Por lo tanto, no se pueden usar las IQMs para detectar información en imágenes de las que solo se dispone una copia. Aunque si otro tipo de medidas, como las estadísticas de primer orden [53, 54]: media, varianza, desviación típica, curtosis etc. Ya que estas se pueden calcular a una única imagen. Además, otro aliciente es que se pueden usar clasificadores, para los cuales existen más variedad de algoritmos que para las técnicas de clusters. Debido al uso de clasificadores hay que clasificar cada imagen con una clase, según tenga información oculta o no.

En los artículos estudiados se observa que la creación de estegoanalizadores universales, en imágenes, usando IA, sigue siempre el mismo proceso. Este proceso, usado en la documentación analizada, se resume a continuación.

1. **Conjunto de Imágenes**: creación de un conjunto de imágenes con y sin información oculta (usando para ello herramientas esteganográficas).
2. **Extracción de Medidas**: cálculo de una serie de medidas por cada imagen del conjunto.
    a. **IQM:** si se dispone de pares de imágenes a la hora de la evaluación.
    b. **Estadísticas de primer orden:** si se dispone únicamente de una imagen en el momento de la evaluación.
3. **Creación de Instancias**: creación de un conjunto de instancias de entrenamiento.
4. **Entrenamiento con IA**: entrenamiento de algoritmos de aprendizaje automático usando o no una previa selección de los mejores atributos.
5. **Evaluación**: se prueba el estegoanalizador construido para comprobar su eficacia.

Esta información sobre la creación de estegoanalizadores con IA, se infiere y extrapola al problema para crear un marco genérico de creación de estegoanalizadores. Este marco será descrito en la sección 3.2 (Descripción General del Marco). El siguiente paso, será describir estos pasos en un marco general, para el problema a solucionar y concretar las siguientes cuestiones:

- El tipo de herramientas esteganográficas.
- Las medidas a extraer.
- Las clases de las instancias.
- El tipo de selección de atributos a usar.
- El tipo de clasificadores a usar.

Como se muestra en la literatura, casi todas las opciones que hay de IA son dirigidas a algoritmos específicos, pero que no mencionan ningún método para crear estego analizadores de forma genérica. Además, de los estegoanalizadores creados en los experimentos, sólo se

han publicado sus resultados y no un programa o al menos código fuente para poder utilizarlos.

3.1.2.2 Herramientas del Estegoanálisis

A continuación, antes de proceder a analizar el marco general, se enumeran ciertas herramientas software estegoanalíticas de interés para su análisis y búsqueda de características aplicables al problema a resolver.

- **StegDetect [**57**]:** Es una herramienta software que detecta contenido esteganográfico en imágenes, de forma automática. Es capaz de detectar diferentes métodos esteganográficos para embeber información oculta en imágenes JPEG. Actualmente las herramientas que detecta son jsteg, jphide, invisible secrets, outguess 01.3b, F5 (con análisis de cabeceras [58, 59]), appendX y camouflage. Usa un modelo, el cual también soporta detección de nuevos estegosistemas. Se usa Stegbreak para lanzar ataques de diccionario contra JSteg-Shell, JPHide, y OutGuess 0.13b.

- **Ben-4D Steganalysis Software [**60**]:** Herramienta de código abierto, desarrollada por Alex Zaharis y Di-Mar. Realiza una rápida y precisa identificación sobre los archivos de soporte esteganográfico, de una colección de archivos. Se aplica una generalización de los principios básicos de la distribución de la Ley de Benford[14], sobre el archivo sospechoso a fin de decidir si el archivo tiene información oculta.

- **Digital Invisible Ink Toolkit (DIIT) [**45**]:** Este proyecto puede ocultar mensajes en imágenes de color de 24 bits, así como de saber si esta tiene información oculta o realizar análisis estadísticos.

- **StegSecret [**61**]:** Es un proyecto de software libre que tiene la intención de desarrollar y mantener un conjunto de herramientas libres, que permitan la detección de información esteganografiada en diferentes medios de información. Se basa en la detección de patrones fijos que dejan las herramientas esteganográficas.

- **StegAlyzerAS (Steganography Analyzer Artifact Scanner) [**62**]:** Desarrollado por SARC (Steganography Analysis and Research Center). Es una herramienta de pago, aunque se puede descargar una versión de evaluación. Detecta archivos y entradas del registro asociadas con aplicaciones esteganográficas.

- **StegAlyzerSS (Steganography Analyzer Signature Scanner) [**62**]:** Desarrollado también por SARC (Steganography Analysis and Research Center). Es también de pago, aunque se puede descargar una versión de evaluación. Detecta archivos esteganográficos y extrae la información de estos.

[14] La ley de Benford, también conocida como la ley del primer dígito, asegura que, en los números que existen en la vida real, la primera cifra es 1 con mucha más frecuencia que el resto de los números.

- **StegAlyzerRTS (Steganography Analyzer Real-Time Scanner) [**62**]:** Desarrollado también por SARC (Steganography Analysis and Research Center). Es también de pago, aunque se puede descargar una versión de evaluación. Detecta artefactos y firmas esteganográficas en tiempo real sobre la red.

- **Stego Watch [**63**]:** Desarrollado por Stego Suite. Es un programa de pago. Detecta información oculta en imágenes y ficheros de audio.

- **Stego Analyst [**63**]:** Desarrollado también por Stego Suite. Es también un programa de pago. Permite el análisis de propiedades de imágenes como el brillo, el contraste, la paleta de colores, los valores RGB de los píxeles, analizar histogramas de coeficiente DCT.

- **Stego Break [**63**]:** Desarrollado también por Stego Suite. Es también un programa de pago. Diseñado para recuperar la clave utilizada en el proceso de codificación de ciertos algoritmos como F5, JPHide&Seek, Camouflage y Jsteg.

- **Stego Hunter [**63**]:** Desarrollado también por Stego Suite. Es también un programa de pago. Una serie de escaneadores activos orientados para detectar la presencia de información oculta en una web concreta o en un dominio corporativo.

3.2 Descripción General del Marco

El objetivo del proyecto, es la creación de un marco único para la creación de estegoanalizadores de imágenes usando herramientas esteganográficas, extracción de medidas e inteligencia artificial. Después de haber estudiado la literatura existente se han generalizado todas las técnicas de creación de estegoanalizadores para imágenes, para desarrollar un marco único que las recoja todas.

En esta sección se hará una descripción general del marco a diseñar posteriormente, donde se describirá de forma general como se pretende obtener un estegoanalizador para cualquier algoritmo esteganográfico. Esta descripción va a permitir especificar en la siguiente sección un análisis de las funcionalidades que debe incluir la aplicación a diseñar. Se propone el siguiente procedimiento para la creación de un estegoanalizador universal:

1. **Obtención de Imágenes**: creación de una base de datos de imágenes, que se divide en dos:
    a. Imágenes “limpias”, sin información oculta.
    b. Imágenes con información oculta, usando para ello una batería de **herramientas esteganográficas** que cumplan las siguientes características.
        i. Usen varios métodos de ocultación (LSB, DCT, DFT, DWT).
        ii. Usen varios formatos de entrada (BMP, JPG o JPEG, GIF, PNG).
2. **Extracción de Medidas**: cálculo de una serie de medidas por cada imagen. Se han decidido agrupar en los siguientes conjuntos:
    a. **Aleatoriedad:** tests estadísticos para analizar la aleatoriedad en las imágenes.
    b. **Estadísticas de primer orden:** estadísticos como la media, la varianza, etc.
    c. **Estegoanalíticas:** basadas en técnicas de detección de información oculta.
    d. **Otras Características:** otras medidas relacionadas con la propia imagen, como el formato.
3. **Creación de Instancias**: las medidas extraídas servirán para las técnicas de la inteligencia artificial. Cada medida calculada de una imagen se convertirá en un atributo de una instancia. Habiendo una instancia por cada imagen. Cada instancia, de una imagen, tiene una serie de atributos. Y el último atributo será una clase, la que indica si la imagen tiene información oculta o no.
4. **Entrenamiento con IA**: entrenamiento de diferentes técnicas de la inteligencia artificial.
5. **Evaluación**: se prueba el estegoanalizador construido para comprobar su eficacia, con nuevas imágenes.

6. **Generación del Estegoanalizador final**: por último se genera un estego analizador que es capaz de analizar cualquier imagen directamente.

Este marco podrá usarse indefinidamente y de distintas maneras, para crear los estegoanalizadores que se deseen, pudiéndose crear, por ejemplo, analizadores de una única técnica esteganográfica o de varias.

En resumen, este marco permite la generación de un modelo que se pueda aplicar directamente a nuevos conjuntos de imágenes, consiguiéndose de esta manera una forma fácil de desplegar los posibles estegoanalizadores. Por ejemplo, en caso de que quisiesen ser implementados en empresas para protegerse ante fugas de información, o en estudios para medir la resistencia de nuevas herramientas esteganográficas, o crear nuevos estegoanalizadores, etc.

3.3 Análisis de Funcionalidades

En este apartado se van a analizar las funcionalidades de la descripción general del marco. Se van a desarrollar los casos de uso, los cuales permiten identificar los requisitos funcionales que necesita el sistema. Es importante destacar que, debido al carácter investigador de la aplicación, se ha decidido no incluir un análisis con un catálogo de requisitos de usuario ni del software. Aunque se desarrollarán los modelos de caso de uso para ayudar a la extracción de funcionalidades y poder aplicar estas en el diseño del software.

3.3.1 Modelo de casos de uso

El usuario potencial de la aplicación es un investigador en el campo de las Ciencias de la Computación e Inteligencia Artificial. Los modelos de casos de uso se muestran en el siguiente diagrama de la Figura 9:

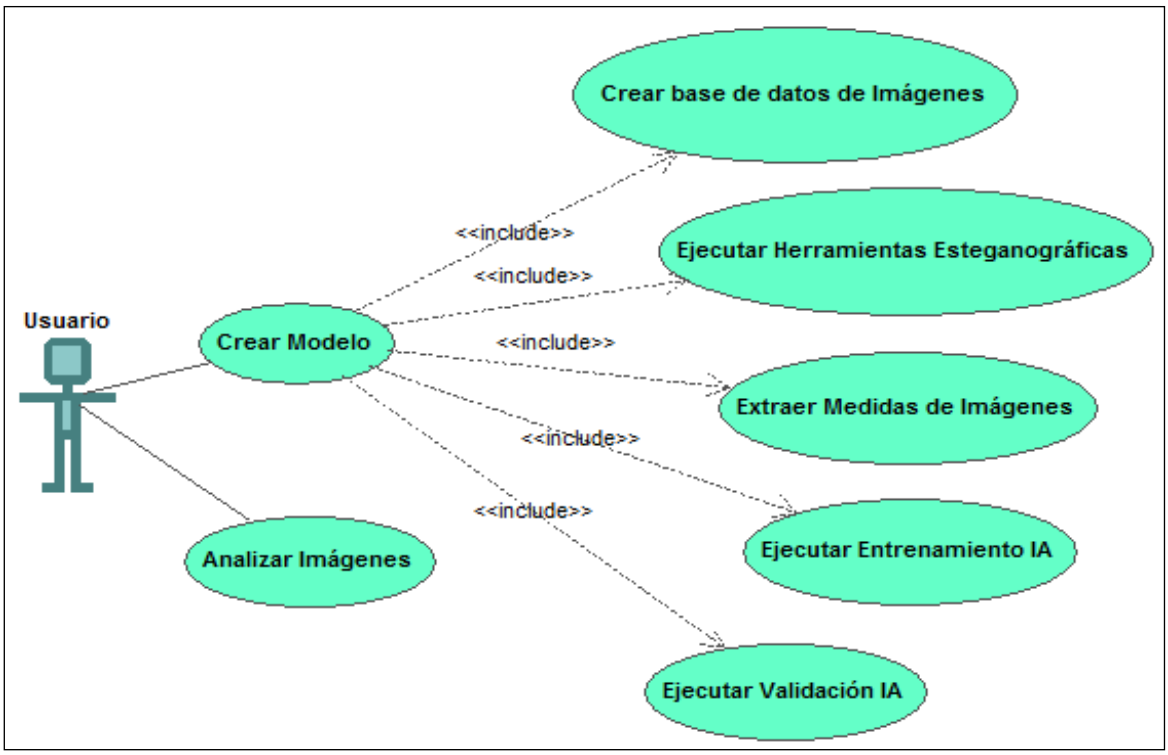

Figura 9: Modelo de casos de uso

El usuario dispone de dos funcionalidades de primer nivel:

- **Crear Modelo**: El usuario puede crear un modelo, llevando a cabo todos los casos de uso que incluye el caso de Crear Modelo. Esta inclusión implica el uso de *include*, ya que se describe un comportamiento común reutilizable en el caso de Crear Modelo. Aunque estos pueden realizarse en momentos distintos, pero habrá que realizarlos todos para llegar a crear el modelo. El modelo, es un clasificador entrenado con técnicas de IA, que el usuario puede validar para predecir si hay información oculta o no, desde el caso de uso Ejecutar Validación IA.

- **Analizar Imágenes**: El usuario puede analizar imágenes independientemente de si ha creado un modelo previamente. La funcionalidad asociada a este caso es analizar imágenes como cualquier herramienta existente de estegoanálisis y no tiene relación con la creación de modelos.

3.3.2 Especificación Textual de los Casos de uso

Cada caso de uso se especifica textualmente, donde se explican los pasos que lleva el usuario para realizar estos. Esta información viene dada primordialmente por el marco descrito en la sección 3.2. Además, esta definición ayudará a extraer información concreta que luego será usada en el diseño y en la implementación.

Nombre	CU 1 – Crear Modelo
Actores	Usuario
Objetivo	El usuario crea un modelo de un estegoanalizador
Precondiciones	El usuario dispone de un conjunto de imágenes
Postcondiciones	Un modelo nuevo es creado
Escenario básico	1. El usuario realiza cada parte de la creación por orden: • Crear base de datos de imágenes • Usar Herramientas Esteganográficas • Extraer Medidas de Imágenes • Ejecutar Entrenamiento IA • Ejecutar Validación IA 2. El usuario guarda el modelo 3. El usuario finaliza la creación del modelo
Escenario alternativo	1-3a. El usuario no sigue el orden correcto para la creación 2. El usuario finaliza la creación del modelo
	1-3a. El usuario realiza cada parte de la creación por orden 2. El usuario no guarda el modelo 3. El usuario finaliza la creación del modelo

Tabla 4: Caso de uso 1 - Crear Estegoanalizador

Nombre	CU 2 – Analizar Imágenes
Actores	Usuario
Objetivo	El usuario analiza imágenes para buscar información oculta
Precondiciones	El usuario dispone de imágenes a analizar
Postcondiciones	Se informa de las imágenes con información oculta
Escenario básico	1. El usuario carga una serie de imágenes 2. El usuario espera al análisis 3. El usuario comprueba los resultados del análisis 4. El usuario finaliza el análisis de imágenes
Escenario alternativo	1-3a. El usuario no carga ninguna imagen 2. El usuario finaliza el análisis de imágenes

Tabla 5: Caso de uso 2 – Analizar Imágenes

Nombre	CU 3 – Crear base de datos de Imágenes
Actores	Usuario
Objetivo	El usuario crea una base de datos con imágenes
Precondiciones	El usuario dispone de un conjunto de imágenes
Postcondiciones	Una base de datos con imágenes es creada
Escenario básico	1. El usuario carga una serie de imágenes 2. El usuario añade las imágenes a una base de datos 3. El usuario guarda la base de datos 4. El usuario finaliza la creación de la base de datos
Escenario alternativo	1-3a. El usuario no carga ninguna imagen 2. El usuario finaliza la creación de la base de datos

Tabla 6: Caso de uso 3 - Crear base de datos de Imágenes

Nombre	CU 4 – Ejecutar Herramientas Esteganográficas
Actores	Usuario
Objetivo	El usuario aplica una o varias herramientas esteganográficas a una base de datos de imágenes
Precondiciones	El usuario dispone de una base de datos de imágenes
Postcondiciones	Se crea una base de datos con imágenes, con información oculta, por cada herramienta ejecutada
Escenario básico	1. El usuario carga una base de datos de imágenes 2. El usuario elige la(s) herramienta(s) a usar 3. El usuario ejecuta la(s) herramienta(s) 4. El usuario guarda la(s) base(s) de datos 5. El usuario finaliza el uso de la(s) herramienta(s) esteganográfica(s)
Escenario alternativo	1-3a. El usuario no carga ninguna base de datos 2. El usuario finaliza el uso de la(s) herramienta(s) esteganográfica(s)
	1. El usuario carga una base de datos de imágenes 2-4a. El usuario no elige la(s) herramienta(s) a usar 3. El usuario finaliza el uso de la(s) herramienta(s) esteganográfica(s)

Tabla 7: Caso de uso 4 – Ejecutar Herramientas Esteganográficas

Nombre	CU 5 – Extraer Medidas de Imágenes
Actores	Usuario
Objetivo	El usuario calcula medidas para cada imagen de una base de datos de imágenes
Precondiciones	El usuario dispone de una base de datos de imágenes
Postcondiciones	Se crean los ejemplos o instancias necesarios para la IA
Escenario básico	1. El usuario carga una base de datos de imágenes 2. El usuario elige las medidas a extraer 3. El usuario ejecuta la extracción 4. El usuario guarda las instancias 5. El usuario finaliza la extracción de medidas
Escenario alternativo	1-3a. El usuario no carga ninguna base de datos 2. El usuario finaliza la extracción de medidas
	1. El usuario carga una base de datos de imágenes 2-4a. El usuario no elige las medidas a extraer 3. El usuario finaliza la extracción de medidas

Tabla 8: Caso de uso 5 – Extraer Medidas de Imágenes

Nombre	CU 6 – Ejecutar Entrenamiento IA
Actores	Usuario
Objetivo	El usuario aplica una o varias técnicas de IA sobre unas instancias
Precondiciones	El usuario dispone de instancias
Postcondiciones	Se crea un modelo entrenado con las instancias
Escenario básico	1. El usuario carga unas instancias 2. El usuario elige la(s) técnica(s) de IA a usar 3. El usuario ejecuta la(s) técnica(s) 4. El usuario guarda el modelo resultante 5. El usuario finaliza la ejecución del entrenamiento de IA
Escenario alternativo	1-3a. El usuario no carga ninguna instancia 2. El usuario finaliza la ejecución del entrenamiento de IA
	1. El usuario carga unas instancias 2-4a. El usuario no elige la(s) técnica(s) de IA a usar 3. El usuario finaliza la ejecución del entrenamiento de IA

Tabla 9: Caso de uso 6 – Ejecutar Entrenamiento IA

Nombre	CU 7 – Ejecutar Validación IA
Actores	Usuario
Objetivo	El usuario valida un modelo creado frente a una instancias
Precondiciones	El usuario dispone de un modelo y de instancias
Postcondiciones	Se muestran unas predicciones del modelo sobre las instancias
Escenario básico	1. El usuario carga un modelo 2. El usuario carga unas instancias 3. El usuario ejecuta la validación 4. El usuario comprueba los resultados de la validación 5. El usuario finaliza la ejecución de la validación del modelo
Escenario alternativo	1-4a. El usuario no carga ningún modelo 2. El usuario finaliza la ejecución de la validación del modelo
	1. El usuario carga un modelo 2-4a. El usuario no carga unas instancias 3. El usuario finaliza la ejecución de la validación del modelo

Tabla 10: Caso de uso 7 – Ejecutar Validación IA

4. DISEÑO

En esta sección se explicará el diseño general del marco, descomponiendo y describiendo y especificando sus partes:

4.1 Descomposición de los módulos principales del Marco

A continuación, se explica el marco de forma más detallada, explicando a su vez ciertos conceptos claves que tienen que ver con su definición. Se incluyen referencias a los casos de uso de partida de cada módulo.

1. En primera instancia, se va a trabajar con cuatro formatos de imágenes (JPG, BMP, GIF, PNG). Como se espera usar gran cantidad de imágenes ubicadas en sitios diferentes, se deberá crear una entidad que aglutine todas estas imágenes en una sola referencia. A esta entidad se la denominará conjunto de imágenes. Este conjunto será el punto de partida de la aplicación, y a su realización se la llamará *Generación de Conjuntos de Imágenes.* Este módulo es la funcionalidad indicada con el CU 3.

2. En segundo lugar, se usará una batería de herramientas esteganográficas (CU 4), para ocultar información en los conjuntos de imágenes creados. Emplearán varios formatos y varias técnicas de ocultación (LSB y DCT, DFT y DWT).

3. Después, como punto intermedio, antes de usar las técnicas de Aprendizaje Automático, se deben extraer características de las imágenes de los conjuntos de imágenes con los que se quiere trabajar (CU 5). Estas medidas serán después los atributos que representen a cada imagen en los algoritmos de la inteligencia artificial. Esta tarea se denomina *Extracción de Medidas de Imágenes*

4. Se usarán las técnicas de Aprendizaje Automático, que se denominará como fase de *Entrenamiento* (CU 6) *y Test* (CU 7). En el entrenamiento el clasificador aprende, construyéndose así el analizador**.** En el test, el clasificador se evalúa con nuevas imágenes para comprobar su eficacia.

5. Por último, se usará el modelo creado para crear un estegoanalizador propio (CU 2).

Por lo tanto, los elementos o módulos principales del diseño general del marco son las mostradas en la Figura 10:

Figura 10: Elementos principales del marco

4.2 Descripción de los módulos principales del Marco

En este punto se describirá cada módulo del apartado anterior, definiendo también brevemente que entradas posee, que proceso realiza, y que salida genera.

4.2.1 Generación de Conjuntos

Este módulo es el encargado de, crear conjuntos de imágenes que luego serán utilizados, según la Figura anterior, para la ocultación de información, usando herramientas esteganográficas. Gracias a estos conjuntos, por ejemplo, se podrá comparar cómo se comportan diferentes herramientas esteganográficas con el mismo conjunto.

Este elemento también es capaz de convertir conjuntos de imágenes con unos formatos a otros, y probar así con herramientas esteganográficas que sólo funcionan con unos formatos limitados. También se podrán transformar las imágenes usando DCT, DFT y DWT, para desarrollar estegoanalizadores que aprendan a detectar las imágenes, con información oculta por herramientas esteganográficas que usan estas transformadas.

La generación de conjuntos recibe como Entrada una o varias imágenes, ubicadas en rutas iguales o distintas, con formatos iguales o distintos. El proceso que realiza es de agrupación, de todas las imágenes que se quieren incluir en el conjunto. Para ello recopila información de cada una de las imágenes. Esta información incluye referencias que permiten saber donde se encuentra cada imagen, para poder trabajar después con ellas.

La salida del proceso será una instancia, que aglutine toda la información recopilada, y que permita recuperar cada imagen, con sólo leer esta salida. Esta instancia es un fichero con un formato estandarizado, y una estructura jerarquizada. De esta forma, será más fácil para los usuarios finales trabajar con esta salida.

La generación de conjuntos permite crear varios conjuntos, habiendo por cada conjunto creado un fichero. En la Figura 11 se puede ver la entrada y la salida del proceso, de forma resumida:

Figura 11: Proceso de la generación de conjuntos

4.2.2 Herramientas Esteganográficas

El siguiente módulo, se encarga de ejecutar una serie de herramientas esteganográficas para la ocultación de información en conjuntos de imágenes. Gracias a esta, se puede automatizar el proceso de ocultación. Seleccionando además que herramientas se quieren usar para cada conjunto.

Este módulo recibe como entrada un conjunto de imágenes, creado previamente con la generación de conjuntos. El proceso que realiza esta tarea es ocultar información, con una batería de programas esteganográficos, en todas las imágenes que hacen referencia el conjunto entrada.

Como salida se genera otro conjunto de imágenes, pero esta vez hace referencia a las imágenes con información oculta. Así que, este proceso genera dos salidas: las imágenes con información oculta, y el fichero *XML* que representa a las imágenes esteganografiadas.

En la Figura 12 se puede ver la entrada y la salida del proceso, de forma resumida:

Figura 12: Módulo de las herramientas esteganográficas

4.2.3 Extracción de Medidas

Este módulo extraerá una serie de características para cada imagen de un conjunto dado. Esta serie de características formará los atributos de una instancia. Por cada imagen se tiene una instancia con sus atributos. Es decir, por cada imagen se tendrá una serie de números que dan información de dicha imagen. Se crearán todas las instancias y esta información la usarán los clasificadores, en el entrenamiento, para aprender que imágenes tienen información oculta o no.

Este módulo tiene como entrada un conjunto de imágenes, con información oculta o sin ella, según desee el usuario extraer medidas de un tipo de imagen u otro. El proceso, que realiza este módulo, es el cálculo de características de las imágenes del conjunto de entrada.

La salida es un fichero de instancias o patrones que se usará para los algoritmos de aprendizaje automático. Se ha decidido utilizar los algoritmos proporcionados por la herramienta Weka [64]. El fichero de instancias con el que trabaja Weka se explicará en detalle en el apartado de Implementación.

En la Figura 13 se puede ver la entrada y la salida del proceso, de forma resumida:

Figura 13: Proceso de la extracción de medidas

4.2.4 Entrenamiento y Test

En este módulo se implementa la inteligencia artificial. Usa instancias que analizan ciertos clasificadores, para llegar a inferir la diferencia entre imágenes con información oculta y sin ella. Para esto son necesarias instancias extraídas de conjuntos de imágenes con información oculta y sin ella.

Dependiendo de las configuraciones de las anteriores etapas se pueden construir modelos distintos, es decir, clasificadores que ya han aprendido, para romper una herramienta esteganográfica en concreto, o intentar generalizar para que el analizador sepa detectar información oculta embebida con cualquier herramienta.

Se tiene como entrada uno o varios ficheros de patrones**,** con instancias de imágenes con información oculta y sin ella, para que los algoritmos de aprendizaje automático aprendan estas instancias, generalicen, y produzcan unos modelos que después del entrenamiento se puedan evaluar (test).

El proceso consiste en que dados unos ficheros de instancias, realizar una serie de experimentos, los cuales generen unos modelos (clasificadores entrenados) cuya ejecución sería como ejecutar un estegoanalizador y que se guardan un fichero, cuyo formato lo decide Weka. Estos modelos, y su posterior evaluación, serán la salida, de la tarea de experimentación.

En la Figura 14 se puede ver la entrada y la salida del proceso, de forma resumida:

Figura 14: Proceso de entrenamiento y test

4.2.5 Estegoanalizador de Imágenes

Este módulo permite analizar imágenes usando el mejor estegoanalizador, resultado de una serie de experimentos. Como se puede comprobar en la Figura 15, como entrada recibe una serie de imágenes, las cuales son analizadas mostrando como salida únicamente las imágenes que dieron positivo en el análisis.

Figura 15: Proceso de ejecución de estegoanalizador

4.3 Especificación de los módulos principales del Marco

En esta sección se especificará el diseño descrito en la sección anterior. Se incluirán modelos y diagramas para mostrar la estructura del diseño de la aplicación.

4.3.1 Generación de Conjuntos

Se compone de tres componentes: una principal, y dos complementarias, cuyo objetivo es añadir funcionalidades a la herramienta y una mayor gama de posibilidades para los usuarios. La Figura 16 representa los componentes de la generación de conjuntos:

Figura 16: Componentes de la generación de conjuntos

4.3.1.1 Creación de Conjuntos

Esta es la parte principal y más importante de la generación de conjuntos. Es donde realmente se crean los conjuntos a partir de imágenes seleccionadas por el usuario. En la Figura 17 se puede ver un diagrama explicativo de las funciones que se dan en la creación de conjuntos.

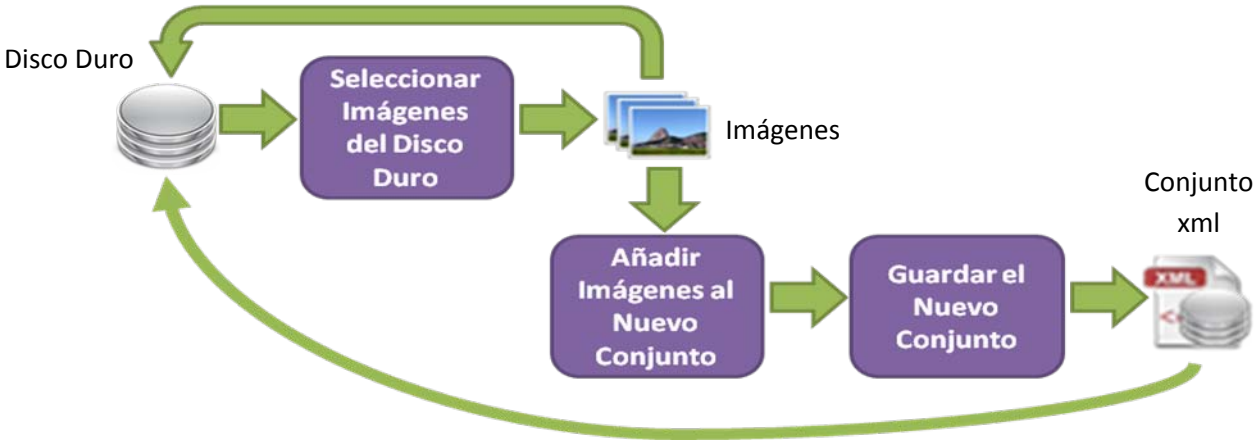

Figura 17: Diagrama de la creación de conjuntos

La primera funcionalidad es seleccionar imágenes del disco duro**.** Las imágenes seleccionadas serán añadidas al nuevo conjunto que se va a crear. La selección se puede repetir las veces que el usuario desee, hasta que desee terminar la creación del nuevo conjunto y se guarde este en un fichero *XML*.

4.3.1.2 Complementos de la Creación de Conjuntos

A continuación, se explicará el diseño de los dos últimos módulos de la creación de conjuntos, que complementan a este.

1. Conversión de Imágenes

Esta funcionalidad es un complemento de la generación de conjuntos. Sirve para convertir, las imágenes de un conjunto dado, de cualquier formato a cualquier otro formato de destino. Esto se hace porque no todas las herramientas esteganográficas que se usarán después funcionan con cualquier formato. Por lo que si se pretende usar unas imágenes de partida, en un formato distinto al de las herramientas que se quieren usar, estas no funcionarán. En estas ocasiones, se tienen que convertir las imágenes.

Las posibles opciones de conversión se pueden ver en la Figura 18:

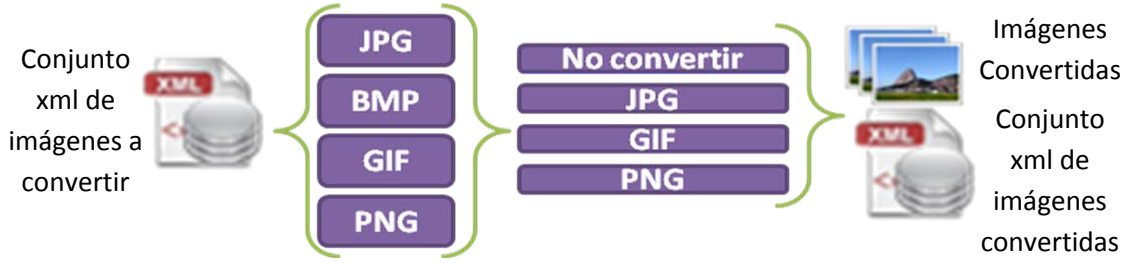

Figura 18: Posibilidades de conversión de las imágenes

Pariendo de un fichero *XML*, de un conjunto de imágenes, que pueden tener cuatro formatos distintos, se puede elegir convertir todas las imágenes de un formato determinado a un formato destino. También se puede elegir convertir sólo algunos formatos y otros no. Como resultado se tendrán las imágenes, a las que hace referencia el conjunto inicial, convertidas en otra ubicación del disco duro. Y para estas imágenes convertidas, se genera un fichero *XML*, que representa al conjunto de imágenes convertidas.

2. *Conversión al Dominio Transformado (DCT, DFT, DWT)*

Este módulo también es complementario y se incluye en la herramienta debido al gran uso de las transformaciones del dominio, en los estegoanalizadores. Como se trató en el capítulo 2, (Conceptos Básicos), la transformación de imágenes se usa en técnicas de ocultación esteganográfica. Debido a esto, también se intentan usar en el estegoanálisis, para poder descubrir esa modificación que se hace en los coeficientes para ocultar información. Hay tres transformadas (DCT, DFT, DWT) y tres tipos más correspondientes a sus inversas (IDCT, IDFT, IDWT). Por lo que la herramienta de conversión tendrá la estructura mostrada en la Figura 19.

Figura 19: Posibilidades de conversión de las imágenes

Partiendo de un conjunto de imágenes, se puede elegir una de las seis transformaciones a aplicar sobre todas las imágenes del conjunto inicial. Como salida se tiene las imágenes transformadas, en la ubicación elegida por el usuario, y un conjunto de las imágenes transformadas.

4.3.2 Herramientas Esteganográficas

Para la implementación del estegoanalizador se ha decidido utilizar siete herramientas esteganográficas de las descritas en el estado del arte (Capítulo 3). Por cada herramienta, e imagen, se ha decidido ocultar un fichero de texto aleatorio. El tamaño de este fichero es en muchos casos calculado por la propia herramienta, que antes de ocultar, se le pregunta cuantos bytes puede ocultar para cada imagen. Algunas herramientas no dicen cuanta información pueden ocultar y en ese caso el usuario puede elegir el tamaño del fichero a ocultar, según la capacidad esteganográfica de la imagen.

$$Capacidad = \frac{CantidadInformación}{TamañoCubierta}$$

Se dan tres tamaños:

- Tamaño Mínimo (Capacidad aproximada del 5% del tamaño de la imagen)
- Tamaño Medio (Capacidad aproximada del 9% del tamaño de la imagen)
- Tamaño Máximo (Capacidad aproximada del 15% del tamaño de la imagen

Estos tamaños dan varias formas de usar la capacidad de la imagen, para que así se pueda estudiar cuanta cantidad de información oculta es detectable. Los porcentajes dados se basan en que dada una típica imagen, con una tasa de inserción baja (alrededor de un 5%) y la ausencia del archivo original, no es posible concluir con certeza que el archivo portador

contiene datos insertados. El porcentaje de inserción aumenta la naturaleza estadística de los coeficientes de la imagen, lo que le diferencia de la normal (imagen sin información oculta) aumentando la sospecha de que contiene información oculta. Por encima del 15% los efectos comienzan a ser visibles a simple vista. El porcentaje del 9% se usa como tamaño medio recomendable para ocultar información. Ya que este está en el punto medio del margen mínimo indetectable (5%) y del margen máximo detectable (15%). A continuación, se enumerarán las siete herramientas esteganográficas utilizadas:

1. **OpenStego v0.5.2** [*24*]
2. **Hide4PGP v2.0** [*26*]
3. **GifShuffle v2.0** [*29*]
4. **F5 v0.1** [*32*]
5. **Steghide v0.5.1** [35]
6. **Blindside v0.9** [*37*]
7. **Vecna v0.6** [*38*]

Se puede ver a continuación un diagrama de clases, en la Figura 20, para el diseño de las herramientas esteganográficas:

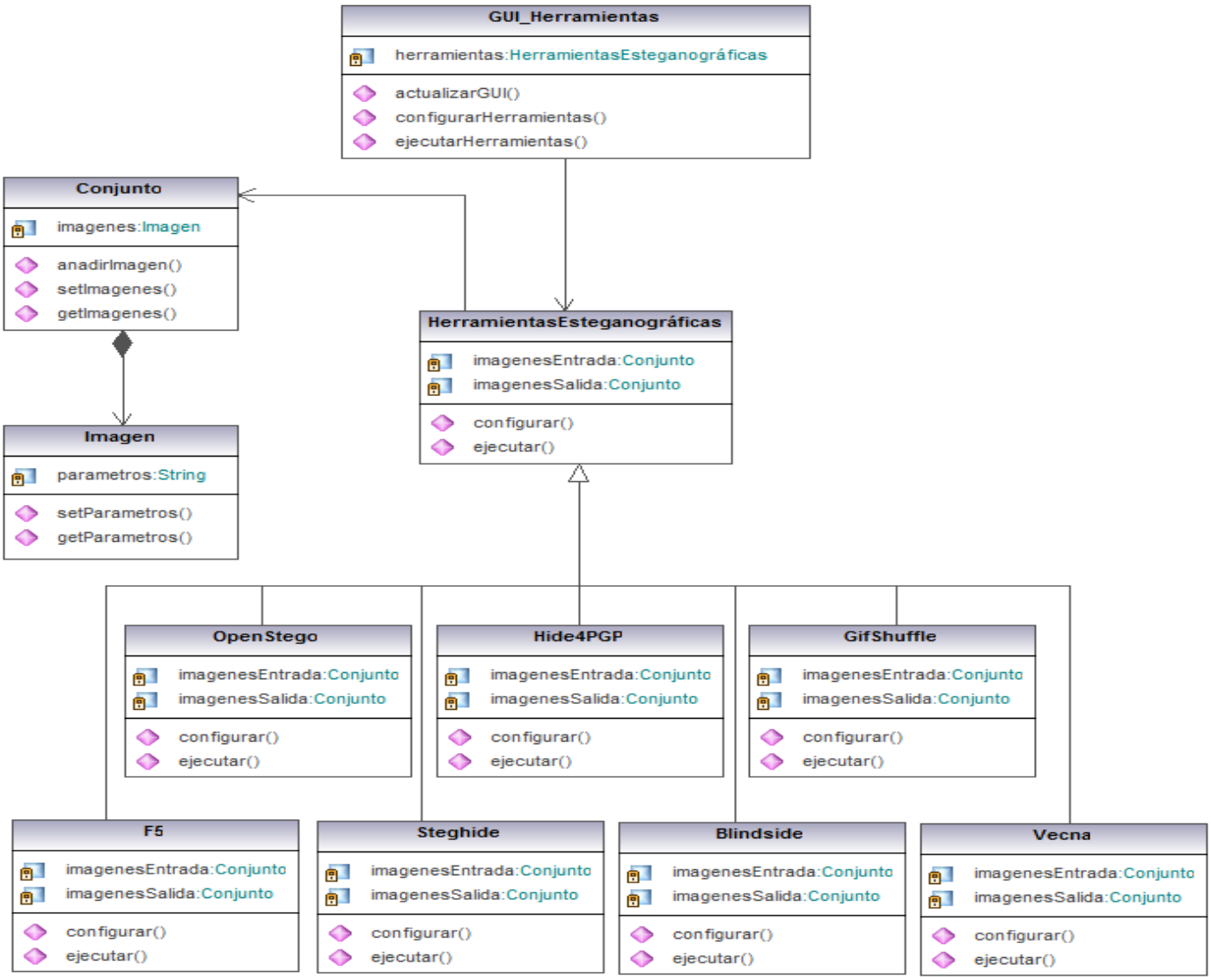

Figura 20: Diagrama de clases de las herramientas esteganográficas

La clase *GUI_Herramientas* se encarga de proporcionar la interfaz de usuario que actúa de intermediario entre el usuario y la aplicación. Esta se va actualizando según cambian sus componentes. Desde ella se puede lanzar la configuración de cada herramienta y su posterior ejecución. La clase principal sería *HerramientasEsteganográficas,* que representa a la aplicación. Es una clase que generaliza el comportamiento de cada una de las siete herramientas. Estas pueden configurarse y ejecutarse sobre unas imágenes de entrada y produciendo otras de salida. Cada herramienta se ejecutará de formas distintas, pero para la clase *GUI_Herramientas* esto será transparente. Las imágenes se agrupan en conjuntos. Esto queda representado con la composición de *Conjunto* e *Imagen*. Donde *Conjunto* está formado por una serie de objetos de la clase *Imagen* que se pueden añadir a este. Cada *Imagen* está formada por una serie de parámetros que permiten identificar y localizar a esta.

4.3.3 Extracción de Medidas

El siguiente módulo, podría considerarse como la fase intermedia de la herramienta. Gracias a él se generarán los ficheros de instancias necesarios para entrenar los algoritmos de aprendizaje automático.

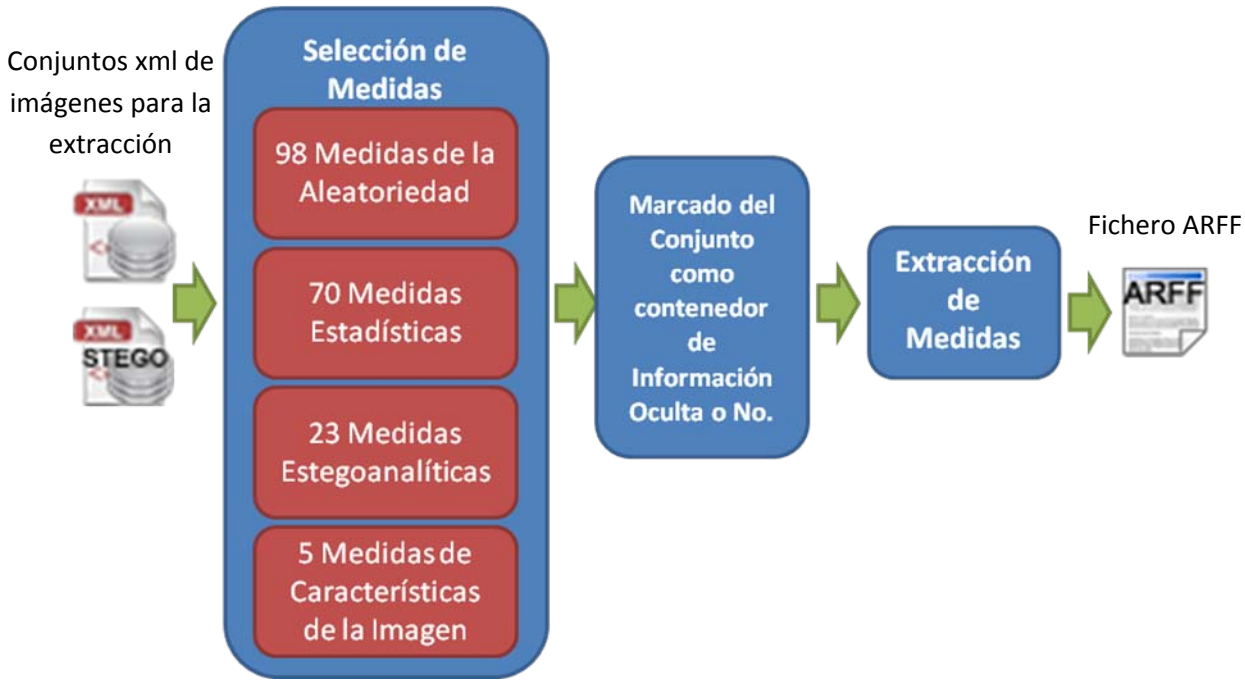

Figura 21: Implementación de la extracción de medidas

Como se observa en el proceso, de la Figura 21, se basa en extraer información derivada del análisis de las imágenes, de cuatro formas (medidas de la aleatoriedad, medidas estadísticas, medidas estegoanalíticas, medidas de características de la imagen), en total se extraen 196 medidas. Después el usuario deberá especificar si el conjunto inicial tiene o no información oculta, para que así se pueda clasificar cada imagen de ese conjunto, donde el último atributo de cada instancia, la clase tomará el valor O ó 1, según no tenga, o tenga, información oculta las imágenes del conjunto. Finalmente se extraen las medidas a un fichero de patrones de Weka [64], con el formato *arff*.

En la siguiente Figura 22 se muestra el diseño de clases para la Extracción de Medidas. En él se encuentra la clase *GUI_ExtraccionMedidas* se encarga de proporcionar la interfaz de usuario que actúa de intermediario entre el usuario y la aplicación. Esta se va actualizando según cambian sus componentes. Desde ella se puede lanzar la selección de las medidas a extraer y su posterior ejecución. La clase principal sería *ExtraccionMedidas,* que representa a la aplicación. Es una clase que generaliza el comportamiento de cada uno de los cuatro grupos de medidas. Estos se ejecutan sobre unas imágenes de entrada y producen un fichero de salida (*arff*) con las instancias formadas por los atributos de las medidas de cada clase. Las imágenes se agrupan en conjuntos. Esto queda representado con la composición de *Conjunto* e *Imagen*. Donde *Conjunto* está formado por una serie de objetos de la clase *Imagen* que se pueden añadir a este. Cada objeto *Imagen* está formada por una serie de parámetros que permiten identificar y localizar a esta. *Arff* representa el fichero de instancia resultante del proceso. Está formado por una serie de parámetros que le identifican y permiten leer y escribir el fichero de instancias.

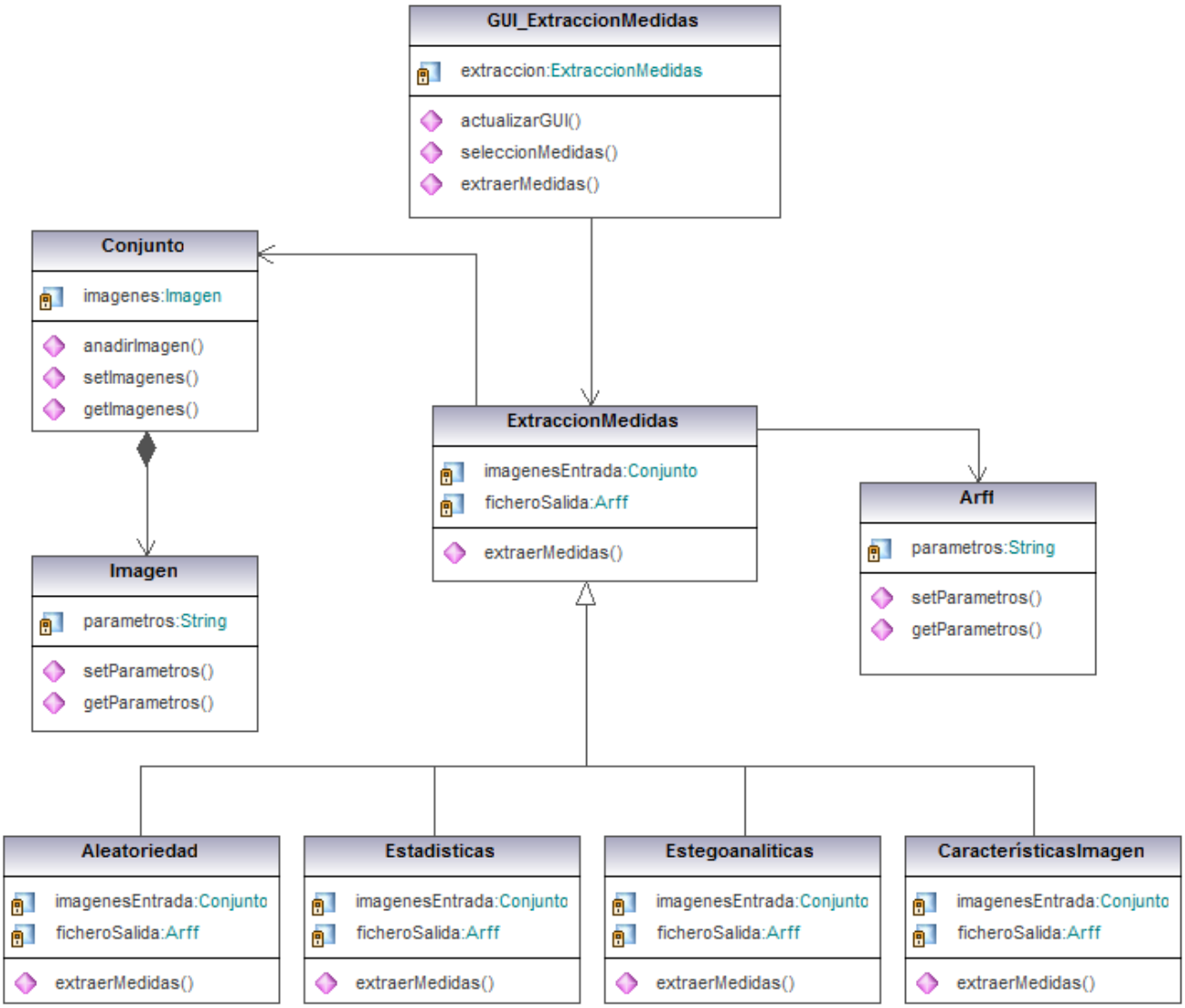

Figura 22: Diagrama de clases de la extracción de medidas

A continuación se enumera cada tipo de medidas:

1 Medidas de la Aleatoriedad:

- Entropía Media por byte
- Porcentaje de Compresión Óptima
- Chi-Cuadrado
- Media Aritmética
- Valor de Monte Carlo para Pi (π)
- Porcentaje del Error del Valor de Monte Carlo
- Coeficiente de Correlación

Para cada imagen se pueden calcular estos tests estadísticos de 14 formas distintas, aunque estas se agruparán en seis grupos, dependiendo a como se apliquen en la imagen. Los seis grupos de las distintas formas en las que se calculan los tests son:

- Imagen entera
- Separando por Canales (RGB)
- Porciones de cada byte
- Separando por Canales (RGB) y Porciones de cada byte
- Juntando Canales (RGB) por pares
- Juntando Canales (RGB) por pares y Porciones de cada byte

En total, hay 14 formas de calcular 7 tests estadísticos (entropía, chi-cuadrado,etc.), por lo que hay en total 98 medidas aleatorias.

2 Medidas Estadísticas:

- Media
- Varianza
- Desviación Típica
- Coeficiente de Variación
- Apuntamiento o Curtosis

Estas medidas de dispersión se calculan de las mismas 14 formas que los tests estadísticos de las medidas aleatorias, por lo que hay un total de 70 medidas estadísticas.

3 Medidas Estegoanalíticas: Hay tres grupos de medidas, con varios valores cada uno:

- Análisis RS (RS Analysis): En total son 14 valores.
- Análisis Sample Pairs (SP Analysis): En total son 8 valores.
- Tasa de Parejas de Color Cerradas/Parejas de Color: Es un único valor.

En total son 23 medidas estegoanalíticas.

4 **Medidas de Características de la Imagen:** Son cinco medidas.

- Formato de la imagen
- Tamaño de la imagen
- Ancho de la imagen
- Alto de la imagen
- Fecha

En total 196 medidas se pueden extraer de una imagen. En el capítulo 5, en la sección 5.3 de Extracción de Medidas, se especificará cada medida enumerada anteriormente, para explicar cómo se calcula cada una.

4.3.4 Entrenamiento y Test

El siguiente módulo es el último de la creación de estegoanalizadores. En él se van a utilizar los algoritmos de aprendizaje automático con los ficheros de instancias de Weka [64] generados en el proceso de extracción de medidas. Para detectar información oculta en imágenes, se usará para las instancias o ejemplos una clase con dos valores (0: si no tiene información oculta y 1: si tiene información oculta). Como son clases discretas, el problema es de clasificación, por lo tanto se usarán clasificadores.

Como se observa en la Figura 23, los ejemplos originales se agruparán en dos grupos (Entrenamiento1 y Test1 (o Evaluación)) y a su vez, Weka, dividirá los de Entrenamiento en Entrenamiento2 y Test2 (o Validación) (de una forma u otra según se use *Percentage Split* o *Cross Validation*):

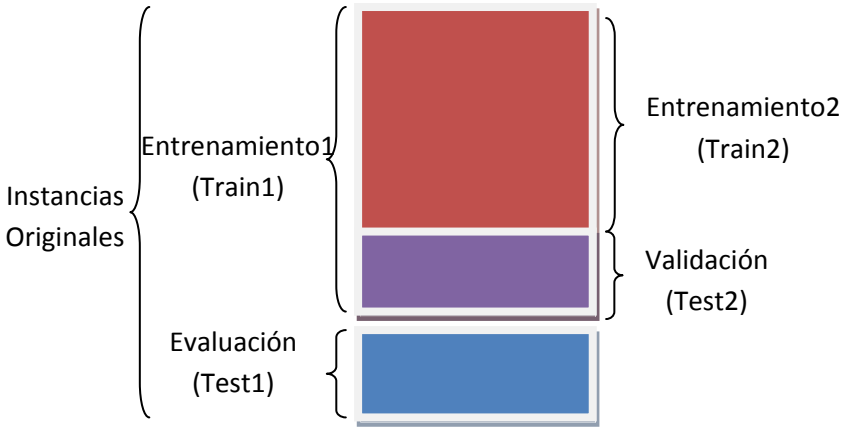

Figura 23: División de instancias de Weka

La diferencia entre Train1 y Test1, es que el clasificador nunca entrenará con los patrones de Test1. Estos se usarán después, una vez obtenido el modelo (clasificador entrenado) para evaluar lo bien que ha generalizado y como clasifica patrones nuevos, desconocidos para él. Weka para conseguir que un clasificador aprenda, divide, a su vez, el conjunto de datos en Train2 y Test2. Donde Test2 son un conjunto de instancias de entrenamiento, las cuales son

usadas por el clasificador para calcular errores y que no se produzca sobreaprendizaje durante el entrenamiento. Una vez entrenados los algoritmos, estos darán unos modelos, que se podrán ejecutar con nuevos ficheros de patrones, para que el modelo prediga las clases de las instancias, de este nuevo fichero, de test.

Se diseñará el entrenamiento y el test, con diversos clasificadores, de forma automática. Para que el usuario sólo tenga que, cargar las instancias, configurar ciertos parámetros del entrenamiento, como el tipo de algoritmo a usar, y finalmente se ejecute de forma automática.

El diseño del entrenamiento y test se compone de varios componentes, como se ve en la Figura 24.

Figura 24: Componentes de la fase de Entrenamiento y Test

A continuación se va a ver cada componente de forma detallada:

4.3.4.1 Creación de Experimentos

Un experimento no es más que una serie de algoritmos de aprendizaje que se van a aplicar a uno o varios ficheros de patrones, usando o no, selección de atributos. Y esta información se va a guardar en un fichero *XML*.

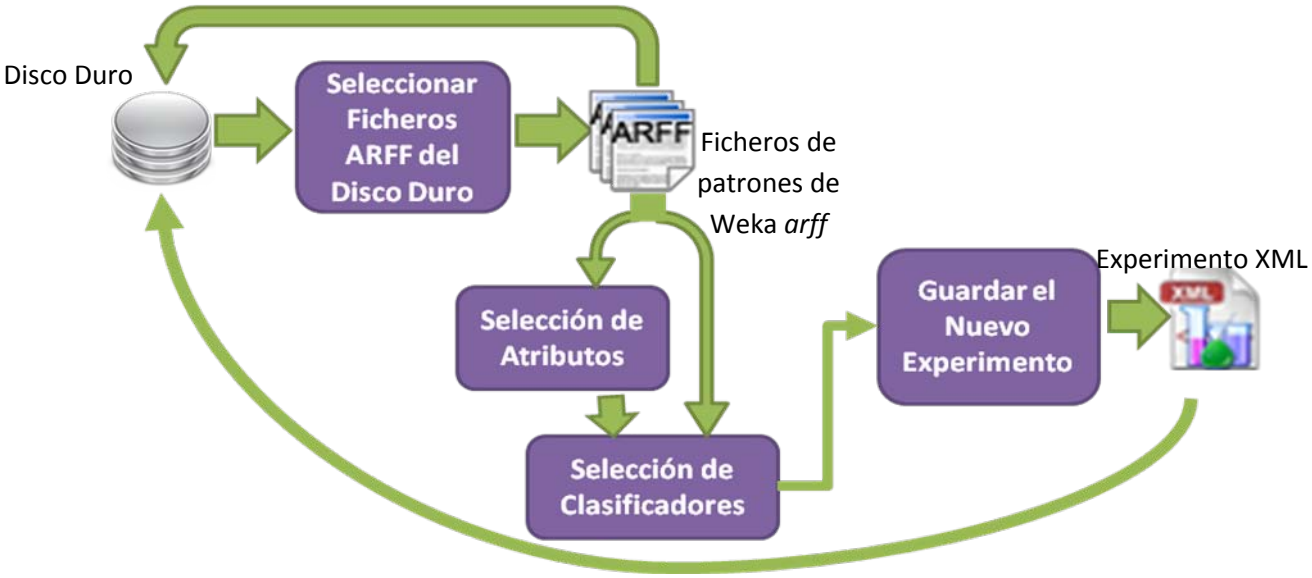

Figura 25: Diagrama de la creación de experimentos

En la Figura 25 se resume el proceso que se explicará a continuación en cuatro pasos. Los cuatro pasos, son los cuatro procesos que están representados en la figura.

1. En primer lugar se seleccionarán los ficheros de instancias (formato *arff*) que desee el usuario para el experimento.

2. Después el usuario podrá, si quiere, realizar una selección de atributos, a cada uno de los ficheros de instancias del experimento que está creando.

3. A continuación, el usuario elegirá los clasificadores que quiera de entre los 12 siguientes:

   - **NaiveBayes**
   - **Logistic**
   - **MultilayerPerceptron**
   - **RBFNetwork**
   - **SMO**
   - **VotedPerceptron**
   - **IB1**
   - **IBk**
   - **KStar**
   - **LWL**
   - **J48**
   - **REPTree**

   Para cada uno de estos clasificadores se puede elegir la técnica de entrenamiento, de entre dos posibles:

   - **Percentage Split (Porcentaje de la división)**
   - **Validación Cruzada**

4. Finalmente el usuario terminará la creación del experimento y podrá guardarlo en un fichero *XML*.

4.3.4.2 Ejecución de Experimentos (Train)

En el módulo anterior se crearon experimentos (formato *XML*) que serán ejecutados en este módulo. Un experimento está formado por uno o más ficheros de patrones (formato *arff*), selección de atributos, o no, y uno o más clasificadores. Después se ejecutará cada experimento creado (formato *XML*) de uno en uno, para que los clasificadores elegidos entrenen. Dando como salida los modelos entrenados (formato *model* de Weka) y los resultados (formato de texto) del entrenamiento.

4.3.4.3 Ejecución de Modelos (Test)

Finalmente, una vez se han ejecutado los experimentos y se han obtenido los modelos, se pueden ejecutar estos contra nuevos ficheros de instancias para evaluarlos y ver como se

clasifican las instancias. En primer lugar se cargará el fichero con el modelo (formato *model*) que se quiere evaluar, y el fichero con las nuevas instancias (formato *arff*) que se quieren evaluar viendo como se clasifican. Después se ejecutará el modelo, y este dará dos salidas en ficheros de texto TXT. Una es la predicción, que hace el modelo para cada instancia. Y la otra es información relativa a los resultados de la evaluación.

4.3.5 Estegoanalizador de Imágenes

Ya que se ha elaborado una herramienta que implementa un marco capaz de crear estegoanalizadores, el siguiente paso lógico es usar esta para crear el mejor estegoanalizador posible y posteriormente usarlo en el propio programa, para que la herramienta pueda ser usada además para analizar imágenes. Se parte de la búsqueda de imágenes y después se analizan estas. Para ello se usa el estegoanalizador creado en la fase de experimentación. Este estegoanalizador es un fichero (formato *model*) de Weka. Finalmente se listan sólo las imágenes que han dado resultado positivo en el análisis.

5. IMPLEMENTACIÓN

A continuación, se explicarán los detalles de implementación de los módulos del marco que necesiten más especificación. Se concretará en cada módulo dando sus detalles más técnicos. Destacar que cada módulo se podrán repetir N veces cada vez que se acabe su ejecución. El resultado final de la implementación podrá verse en el manual (Anexo 1) con una extensa explicación.

5.1 Generación de Conjuntos

5.1.1 Creación de Conjuntos

Se van a especificar los tres procesos dados en el diseño, para tratar los detalles de su implementación:

1. El primer proceso es seleccionar imágenes del disco duro**.** Para ello se parte del disco duro (o cualquier otro dispositivo con almacenamiento), desde donde se cargan las imágenes (JPG, BMP, GIF, BMP) eligiendo un directorio. Esta carga puede hacerse de forma recursiva, y por lo tanto se buscará también en los subdirectorios del directorio elegido. Esta búsqueda puede ser útil, por ejemplo, si se quiere cargar todas las imágenes de un disco duro. Además, se puede realizar un filtrado sobre las imágenes encontradas en el directorio. Es decir, una búsqueda selectiva, según las cualidades de las imágenes (nombre, formato, tamaño, fecha y resolución). Incluso, se podrá deshacer el último filtrado realizado, por si el usuario está descontento con la búsqueda realizada.

2. Las imágenes seleccionadas se añaden al nuevo conjunto. La selección se puede repetir las veces que el usuario desee, hasta que desee terminar la creación del nuevo conjunto y su posterior guardado.

3. Por último, después de que el usuario termine la adición de imágenes al nuevo conjunto, podrá guardar el nuevo conjunto en un fichero *XML***.** Se eligió guardar en un fichero con este formato, porque es un formato estandarizado y su estructura arbórea y con etiquetas, lo hacen idóneo para representar una pequeña base de datos que hace referencia a un grupo de imágenes. Para generar el fichero *XML* se usan las librerías JDOM [65] que proveen de una completa solución en Java, para acceder, manipular y extraer datos *XML* desde código en Java.

   Se puede ver, en la Figura 26, que tiene una estructura jerárquica arbórea, donde Conjunto es la raíz. Presenta una cabecera, con los atributos del *XML*. Después se puede ver la etiqueta del nodo padre, y las características de este conjunto (nombre, fecha en milisegundos, nº de imágenes que lo forman, el tamaño de la suma total de todas las imágenes). Después se van sucediendo todas las imágenes del conjunto, las cuales tienen elementos que definen la información relativa a estas, como la ruta, el nombre y el formato para poder recuperarlas del disco duro. El tamaño, la fecha y la resolución son atributos que no sirven para la carga posterior de las imágenes, pero que tienen carácter informativo.

```
<?xml version="1.0" encoding="UTF-8"?>
<Conjunto nombre="gif" fecha="1281543165303" numero_imagenes="4" tamano="11488">
  <Imagen>
    <Nombre>flecha_alante</Nombre>
    <Ruta>C:\Users\javier\Documents\PFC\PFC\imagenes</Ruta>
    <Formato>gif</Formato>
    <Tamano>2826</Tamano>
    <Fecha>1277227542000</Fecha>
    <Ancho>46</Ancho>
    <Alto>46</Alto>
  </Imagen>
  <Imagen>
    <Nombre>flecha_alante2</Nombre>
    <Ruta>C:\Users\javier\Documents\PFC\PFC\imagenes</Ruta>
    <Formato>gif</Formato>
    <Tamano>2915</Tamano>
    <Fecha>1277227542000</Fecha>
    <Ancho>46</Ancho>
    <Alto>46</Alto>
  </Imagen>
.
.
.
</Conjunto>
```

Cabecera e información del Conjunto

Imágenes del Conjunto

Figura 26: Fichero XML de un conjunto de imágenes

5.1.2 Conversión al Dominio Transformado (DCT, DFT, DWT)

En este complemento se calculan tres transformadas (DCT, DFT, DWT) y tres inversas (IDCT, IDFT, IDWT). Para calcular la DCT e IDCT se ha usado una implementación de Stephen Manley [66]. Para DFT, IDFT, se ha empleado código de Christoph Lauer [67]. Y para DWT e IDWT el autor es desconocido.

5.2 Herramientas Esteganográficas

En la fase de diseño se decidió utilizar siete herramientas esteganográficas. En este módulo hay que configurar cada una de forma separada para poder ejecutarla. Pudiéndose configurar todas, o algunas sí y otras no, según el tipo de ejecución que se quiera realizar. Cada una admite como entrada formatos distintos, y genera también formatos distintos, no siempre iguales a la entrada.

Una vez se han configurado todas, se procede a ejecutar todas directamente, pudiendo paralelizarse el proceso y se dispone de varios núcleos de procesamiento y el usuario lo desea. Para conocer el número de núcleos y otra información referente a la memoria RAM libre, se utiliza la librería Sigar [68].

Por defecto, si la herramienta esteganográfica oportuna, dispone de la posibilidad de comprimir el fichero a ocultar, se ha implementado que por defecto comprima. Si la herramienta esteganográfica, puede cifrar el fichero, se ha implementado que por defecto se cifre. Usando para ello una contraseña aleatoria de entre 8 y 50 caracteres, para garantizar cierta seguridad. Además, si la herramienta esteganográfica usa varios algoritmos de ocultación, se usará el de por defecto, que será el más común usado por los usuarios de esta herramienta. En la Figura 27 se muestra un esquema del proceso seguido:

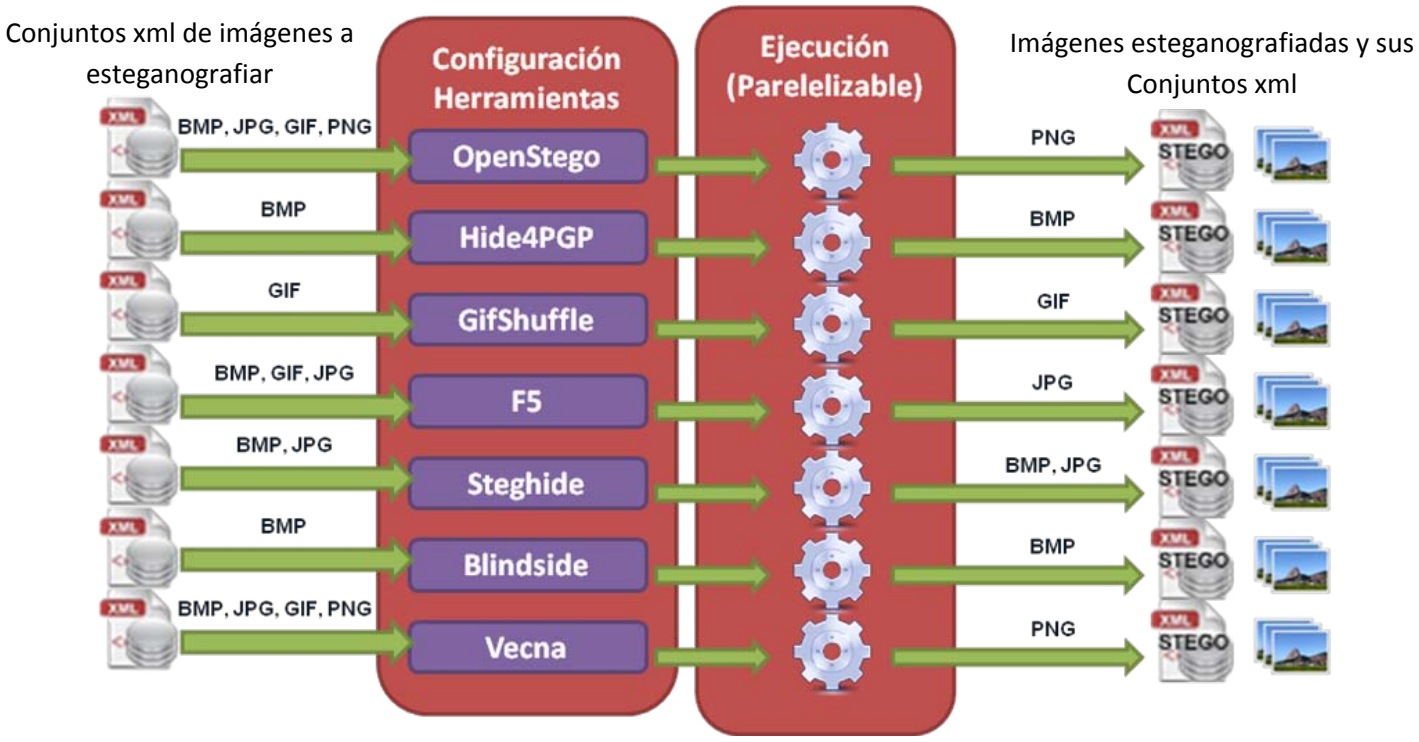

Figura 27: Implementación de las herramientas esteganográficas

Algunas de estas herramientas sólo están disponibles, en su versión ejecutable, en *exe*, lo que hace que sólo se puedan ejecutar en sistemas operativos Windows y por lo tanto el resto de la herramienta se ve afectado por este hecho y restringirá el uso a este tipo de sistema. Por lo

que si de alguna herramienta sólo se disponía el código fuente en ANSI C, este se compilaba para Windows en un *exe*. Si el código fuente es Java, se compilaba en un *jar* para ejecutarse. En alguna ocasión se ha modificado el código fuente de algún programa debido a algún error, o que no se ajustaba a nuestra ejecuciones y ha habido que modificar algún detalle.

5.3 Extracción de Medidas

En este módulo se extrae información relevante de cada imagen, y para ello cada imagen es tratada como un conjunto (secuencia) de bytes y en algunos casos como una secuencia de píxeles. Este conjunto sería equivalente a tener una población de individuos y querer extraer información significante de esta población. Para esto se usan las teorías estadísticas, ya que pueden medir la dispersión de distribuciones de individuos, dando resultados interesantes, de los que a priori no se conoce su existencia.

A continuación se detallará cada tipo de medidas:

5.3.1 Medidas de la Aleatoriedad

Las medidas, aquí descritas, son unos tests estadísticos aplicados, de forma distinta, a las secuencias de bytes que componen los ficheros de imágenes. Estas son calculadas con el programa ENT [69], el cual aplica varios tests a secuencias de bytes de archivos e informa de los resultados obtenidos por estos tests. El programa es usado como una aplicación para evaluar generadores de números pseudoaleatorios para algoritmos de cifrado y aplicaciones estadísticas, algoritmos de compresión y otras aplicaciones donde la información sobre la densidad de los archivos es interesante, como en este caso, en el estegoanálisis.

Estos tests estadísticos son siete:

- **Entropía Media por byte:** Mide la densidad de información del contenido de un archivo, expresado como un número de bits por byte.

- **Porcentaje de Compresión Óptima:** Indica el porcentaje de la imagen que se puede comprimir. Las secuencias aleatorias no se deberían poder comprimir.

- **Chi-Cuadrado:** El test de chi-cuadrado es el más, comúnmente, usado para los test de aleatoriedad de los datos y es extremadamente sensible a errores en los generadores de secuencias pseudoaleatorias. La distribución de chi-cuadrado es calculada para un flujo de bytes en el archivo y expresada como un número absoluto y un porcentaje, el cual indica como una secuencia realmente aleatoria supera el valor calculado. Este porcentaje es interpretado como el grado en que la secuencia de prueba es sospechosa de no ser aleatoria.

- **Media Aritmética:** El resultado de sumar todos los bytes del archivo y dividir por la longitud del archivo. Si los datos están cerca de ser aleatorios, este valor debería de estar en torno a 127,5.

- **Valor de Monte Carlo para Pi (π):** Cada sucesiva secuencia de seis bytes (24 bits) se va disponiendo en un sistema de coordenadas X e Y de forma cuadrada. Cada 24 bits es una coordenada (X ó Y) para formar un punto, que caerá dentro de un área cuadrada. Dentro del cuadrado se dibuja un círculo. Algunos puntos caen dentro del círculo y otros fuera. Se puede calcular la probabilidad (P) de que caiga dentro del círculo. Esta probabilidad es el área del círculo dividido por el área del cuadrado. De lo que resulta $\pi = 4P$. Se va midiendo para cada punto si cae dentro del círculo o fuera. Se van contando los puntos que caen dentro y cuanto más se acerque a **π** significa que es más

aleatoria la secuencia de bytes. Para secuencias muy largas, converge muy lentamente hacia π.

- **Porcentaje del Error del Valor de Monte Carlo:** Esto no es más que el error, medido en porcentaje, de la diferencia que hay entre el valor del test de Monte Carlo y π.

- **Coeficiente de Correlación:** Esta cantidad mide el grado en que cada byte del archivo depende del byte anterior. Para las secuencias aleatorias, este valor (que puede ser positivo o negativo) será cercano a cero.

Para cada imagen se pueden calcular estos tests estadísticos de 14 formas distintas, aunque estas se agruparán en seis grupos, dependiendo a como se apliquen en la imagen. Por la forma en la que trabaja ENT, los test estadísticos se aplican a los bytes del fichero que se le pase como entrada. Pero es importante destacar que, no todos los bytes de una imagen son los píxeles de esta. Ya que una imagen también está compuesta por una cabecera, previa a los datos (píxeles).

En los seis grupos, descritos a continuación, se modificará cada imagen. Pero no se cambiarán directamente los bytes de la imagen, se cambiarán los píxeles, que a su vez cambiarán algunos bytes del fichero. Por lo que ENT, al analizar bytes, detectará estas modificaciones.

Los grupos de las distintas formas en las que se calculan los tests son:

- **Imagen entera:** La imagen original no sufre ninguna transformación antes de calcular los tests. Este grupo sólo tiene una única forma de calcular los tests.

- **Separando por Canales (RGB):** Se puede elegir que componente(s) usar para usar los tests. Para cada uno elegido se construirá una imagen con cada píxel formado sólo con esta componente y las demás a cero. Este grupo tiene tres formas de calcular los tests, una por cada componente.

- **Porciones de cada byte:** Se puede elegir cuantos bits, de cada byte, de cada píxel, usar. Esto se usa porque tiene relación con las técnicas esteganográficas de LSB (bit menos significativo). Este grupo tiene una forma de calcular los tests.

- **Separando por Canales (RGB) y Porciones de cada byte:** Es una mezcla de los dos grupos anteriores. Este grupo tiene tres formas de calcular los tests, una por cada componente.

- **Juntando Canales (RGB) por pares:** Aquí se hacen tres parejas de los tres componentes RGB, que son R-G, R-B y G-B. Se calcularán los tests para cada pareja, teniendo por lo tanto tres formas.

- **Juntando Canales (RGB) por pares y Porciones de cada byte:** Esto es una mezcla de los dos anteriores. No sólo se mezclan los componentes por parejas sino que de los bytes de cada pareja se pueden usar las porciones de bits que se quieran.

En total, hay 14 formas de calcular 7 tests estadísticos (entropía, chi, etc.), por lo que hay en total 98 Medidas Aleatorias.

5.3.2 Medidas Estadísticas

Las medidas aquí descritas, son unas medidas fundamentales de la dispersión, las cuales recogen información sobre conjuntos de datos (o individuos) dispersos, y los representan como si sólo hubiera un único dato (o individuo)**.** En este caso son aplicadas de forma distinta a las secuencias de píxeles de las imágenes. Es decir, cada píxel será como un individuo de una población (siendo la imagen el conjunto de la población), donde se intentará recoger información resumida de toda esta población. Cabe destacar con respecto a las medidas aleatorias, que ahora se calculan para los píxeles de la imagen y no los bytes. Esto es importante, porque todos los bytes de un fichero, que sea una imagen, no son píxeles. Se han seleccionado cinco medidas fundamentales de la dispersión:

- **Media:** Esta es la suma del valor de cada píxel dividido por el número total de píxeles de la imagen.

- **Varianza:** La varianza, S^2, se define como la media de las diferencias cuadráticas de *n* puntuaciones con respecto a su media aritmética, es decir:

$$S^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2$$

- **Desviación Típica:** La desviación típica ***S*** es la raíz cuadrada de la varianza:

$$S = \sqrt{S^2}$$

- **Coeficiente de Variación:** Es la razón entre la desviación típica y la media. Mide la desviación típica en forma de "qué tamaño tiene con respecto a la media":

$$CV = \frac{S}{\bar{x}}$$

- **Apuntamiento o Curtosis:** La curtosis indica el grado de apuntamiento (aplastamiento) de una distribución con respecto a la distribución normal o gaussiana. Es adimensional:

$$K = \frac{1}{S^4 - 3}\sum_{i=1}^{N}(x_i - \bar{x})^4$$

Se resta tres, para generar un coeficiente centrado en 0.

Estas medidas de dispersión se calculan de las mismas 14 formas que los tests estadísticos de las medidas aleatorias.

5.3.3 Medidas Estegoanalíticas

Estas medidas se basan en artículos sobre el estegoanálisis [70]. Dado que estas medidas ayudan a detectar si hay información oculta en imágenes, en procedimientos no automáticos, se comprobará si en procedimientos automáticos, como el que se implementa, ayudan de igual forma.

Hay tres grupos de medidas:

- **Análisis RS (RS Analysis):** es un sistema para detectar técnicas LSB, propuesto por Jessica Fridrich de la Universidad de Binghamton, NY. Explota la correlación espacial en las imágenes con información oculta. Jessica Fridrich, M. Goljan y R. Du, publicaron en octubre de 2001 [71] y [72] un algoritmo muy preciso, de hecho, uno de los más precisos para la detección de LSB-pseudoaleatorio. Su precisión varía en función de la imagen, pero un valor de referencia está en torno a 0'005 bits por píxel (0'5% de la ocupación total posible), para imágenes de alta calidad. Si se oculta más información que 0,005 bits por píxel el algoritmo detectará la presencia de información oculta, y estimará el tamaño de la información enmascarada. Para ello utiliza sensibles estadísticas dobles, derivadas de las correlaciones espaciales en las imágenes.

- **Análisis Sample Pairs (SP Analysis):** es una técnica parar detectar esteganográfia en una imagen [73] de Fridrich. Detecta el bit menos significativo (LSB) en señales digitales tales como imágenes y audio. Puede estimar, con precisión relativamente alta, la longitud del mensaje secreto, ocultado en los bits menos significativos de muestras de señales. Esta técnica se basa en algunas de las medidas estadísticas de las muestras de pares, que son muy sensibles a las operaciones de incrustación en LSB. El algoritmo de detección es sencillo y rápido.

- **Tasa de Parejas de Color Cerradas/Parejas de Color:** Este valor se extrae de la técnica **Raw Quick Pairs** (parejas rápidas brutas) de Fridrich [74] para detectar técnicas LSB. Se basa en que, para imágenes de color verdadero, el número de colores únicos es significativamente más pequeño que el de píxeles. Para imágenes de alta calidad como BMP la relación es 1:2, y de 1:6, más baja, para JPEG. Los colores únicos son los colores de una imagen que al menos salen una vez en ella. Muchas imágenes de color verdadero tienen una pequeña paleta, pero cuando se aplica la técnica LSB, la nueva paleta tiene características distintas al tener muchos colores cerrados. Dos colores son cerrados cuando se diferencian en un bit. Estas características de la paleta son la base de este sistema de detección.

A continuación, se va a explicar los valores que se calculan para cada grupo (RSA, SPA y Tasa de parejas de color):

- **Analisis RS (RS Analysis):** En este análisis se pueden solapar grupos o no. Cada uno de estos dos da 6 valores que son un porcentaje y el número de bytes por cada componente RGB. Además, se hace un promedio del porcentaje y del número de bytes. En total son 14 valores. Para calcular estos valores se ha usado el código fuente DIIT [45], que según la autora, Kathryn Hempstalk, se ha realizado con la ayuda de los autores, del propio análisis RS, para su correcta verificación.

- **Análisis Sample Pairs (SP Analysis):** Aquí se calculan 6 valores que son un porcentaje y el número de bytes por cada componente RGB. Además, se hace un promedio del porcentaje y del número de bytes. En total 8 valores. Para calcular estos valores se ha usado el código fuente DIIT [45], que se ha implementado basándose en el código de C++ de los autores de [73].

- **Tasa de Parejas de Color Cerradas/Parejas de Color:** En primera instancia recordar que, los colores únicos son los colores de una imagen que al menos salen una vez en ella. Y dos colores son cerrados cuando se diferencian en un bit.

  Se tiene que cada color está definido por tres componentes que se llamarán R, G y B, cada una se representa por un byte.

  El número de colores únicos lo se llamarán U. Se dice que dos colores (R_1, G_1, B_1) y (R_2, G_2,B_2) forman una pareja de color cerrada si:

  $|R_1 - R_2| \leq 1$, $|G_1 - G_2| \leq 1$ y $|B_1 - B_2| \leq 1$, o de forma equivalente se puede decir que: $(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2 \leq 3$.

  Considerando sólo los colores únicos, el número de todas las parejas de colores son:

  $$\binom{U}{2} = \frac{U!}{2!(U-2)!}$$

  Se tiene que:

  $$\binom{U}{2} \geq P$$

  Donde P es el número de parejas de color cerrado en la paleta de la imagen. Finalmente se tiene que R, que es el ratio (tasa) entre el número de parejas de color cerradas (P), y todas las parejas de color (U sobre 2).

  $$R = \frac{P}{\binom{U}{2}}$$

  En diversos estudios, R, es calculado por la técnica estegoanalítica *Raw Quick Pairs.* Esta técnica se compone de un algoritmo que usa a R en su funcionamiento. Dicho algoritmo no se ha seguido para desarrollarlo en la herramienta. Sólo se usa la tasa R, de la misma forma en la que se calcula dentro de este algoritmo. La razón de que no se use el algoritmo, es porque el contexto en el que se usa este difiere del contexto del problema que se pretende resolver. El contexto en el que trabaja, es uno tal que, existen parejas de la misma imagen, con información oculta y sin ella. Esto es un caso

muy ideal e irreal, y en la herramienta no se dispone de dos imágenes, sólo de una, y se deberá determinar si tiene información oculta o no. Así que, simplemente se utiliza este ratio, para incluirlo como una medida más.

5.3.4 Medidas de Características de la Imagen

Estas medidas se basan en características muy básicas de las imágenes. La idea es que sean ampliadas en el futuro.

Se usan como complemento a las medidas anteriores, por si las técnicas de aprendizaje infieren una relación entre estas medidas con las anteriores.

Son cinco medidas:

- **Formato de la imagen:** 0 para BMP, 1 para JPG, 2 para GIF y 3 para PNG.
- **Tamaño de la imagen:** Número de bytes de la imagen.
- **Ancho de la imagen:** El número de columnas de píxeles de la imagen.
- **Alto de la imagen:** El número de filas de píxeles de la imagen.
- **Fecha:** La fecha de la última modificación de la imagen. Milisegundos transcurridos desde 1970.

Por último se va a explicar un complemento añadido para usar después de haber extraído medidas de imágenes.

5.3.5 Fusión Ficheros ARFF

El motivo del uso de la fusión es debido a que en la extracción de medidas se extraen ficheros con una única instancia, pero para entrenar a los clasificadores se necesitan ficheros con varias clases, así que esta funcionalidad permite unir ficheros de instancias en uno solo.

Como se ejemplifica en la Figura 28, la fusión de ficheros de patrones (*arff*) carga varios ficheros (del disco duro o cualquier otro dispositivo con almacenamiento), con los mismos atributos, e igual o distinto número de instancias, los fusiona, y como resultado obtiene un fichero, con el mismo número de atributos, y las instancias de los ficheros de instancias cargados.

Figura 28: Diagrama de fusión de ficheros ARFF

Esta fusión es muy útil, ya que se habrán generado ficheros de instancias de varios conjuntos de imágenes, y para ejecutar los algoritmos de aprendizaje automático será deseable tener en un único fichero de patrones con todas las instancias para entrenar o validar.

5.4 Entrenamiento y Test

En esta sección se explicará la implementación del módulo referente al uso de la inteligencia Artificial. Para llevar a cabo el uso de los algoritmos de aprendizaje automático y demás técnicas asociadas a la inteligencia artificial se ha usado Weka. Se hará una descripción de cada componente utilizada de Weka, pero para más información se puede ampliar en [64].

5.4.1 Creación de Experimentos

La creación de experimentos se compone de los siguientes procesos implementables:

5.4.1.1 Selección de ficheros de instancias

En primer lugar se seleccionará del disco duro los ficheros de instancias (o cualquier otro dispositivo con almacenamiento) que desee el usuario para el experimento. Para cada uno de ellos podrá elegir si normalizarlo y/o aleatorizarlo a la hora de la ejecución del experimento en la siguiente subtarea. Normalizar es que cada atributo de cada patrón del fichero de instancias, se cambia a un valor entre 0 y 1, usando la siguiente fórmula:

$$VN_i = \frac{VA_i - VMin_i}{VMax_i - VMin_i}$$

- VN_i = el valor normalizado resultante.
- VA_i = el valor del atributo i-ésimo.
- $VMin_i$ = valor mínimo observado para el atributo i-ésimo.
- $VMax_i$ = valor máximo observado para el atributo i-ésimo.

1. La *normalización* se aplica fundamentalmente cuando el intervalo de los atributos es significativo, y por lo tanto hay valores muy grandes y otros muy pequeños entre los atributos y por lo tanto algunos tienen más peso que otros. Pero también tiene problemas como que todos los atributos (incluso los irrelevantes) tienen el mismo peso, además la clasificación puede ser bastante costosa y es muy sensible a ejemplos ruidosos.

2. *Aleatorizar* implica que se desordenarán las instancias a la hora del entrenamiento de forma aleatoria, es decir, no se preservará el orden inicial en que se encuentran los patrones en el fichero. Esta opción es importante, ya que el fichero de instancias construido en las tareas anteriores, tendrá muchos patrones seguidos con las mismas clases y para entrenar es preferible que el fichero sea heterogéneo, y que no haya sesgos.

5.4.1.2 Selección de atributos

El usuario podrá, si quiere, realizar una selección de atributos, a cada uno de los ficheros de instancias del experimento que está creando. La selección de atributos es útil por varias razones:

- Algunos atributos pueden ser redundantes y hacen más lento el proceso de aprendizaje.
- Algunos atributos pueden ser irrelevantes.
- En ocasiones el exceso de atributos puede llevar al sobreaprendizaje.
- En ocasiones es útil tener el conocimiento de qué atributos son relevantes para una tarea.

Notar que el *sobreaprendizaje* implementa la complejidad del modelo (sobre todo si hay pocos datos). El sobreaprendizaje es un fenómeno que se produce cuando un clasificador obtiene un alto porcentaje de aciertos en entrenamiento, pero pequeño en test, es decir, no generaliza bien. El clasificador está memorizando los datos en lugar de estar generalizando. Se puede detectar también en validación cruzada cuando salen porcentajes cercanos al azar. A continuación, se explicarán en detalle los métodos de la selección de atributos para poder así entender el significado de estos, ya que no son obvios.

5.4.1.2.1 Métodos de Selección de atributos

Los métodos de selección de atributos se dividen en los siguientes grupos. Para más información se puede ampliar consultando en [64].

- **Evaluación individual de atributos** (***Ranker***): se evalúan los atributos de manera independiente. Es muy rápido, pero no detecta atributos redundantes.
- **Evaluación de subconjuntos**: evalúan subconjuntos de atributos.
  - ***Filter***: se evalúan los atributos de los subconjuntos manera individual, pero tiene en cuenta la redundancia entre atributos. Es rápido.
  - ***Wrapper***: se evalúan los atributos de los subconjuntos de manera conjunta. Es lento, pero es el mejor de todos.

El método *Ranker* dado unos atributos, los evalúa cada uno de manera independiente, calculando las medidas de correlación del atributo con la clase. Un atributo está correlacionado con la clase, si conocer su valor implica que se puede predecir la clase con cierta probabilidad. Por ejemplo, el salario de una persona está correlacionado con el hecho de que vaya a devolver un crédito. Los métodos *Filter* evalúan un subconjunto de atributos calculando la media de las correlaciones (o similar) de cada atributo con la clase y descuentan puntos por redundancias entre atributos. Los métodos *Wrapper* evalúan un subconjunto de atributos ejecutando un algoritmo de minería de datos concreto sobre un conjunto de entrenamiento. El valor del subconjunto es el porcentaje de aciertos obtenido con esos atributos. Los algoritmos de los que se dispone son los mismos que se explicarán después en la selección del clasificador. Para los métodos de selección de atributos hay que definir:

- **Búsqueda:** Una manera de moverse por el espacio de búsqueda.
- **Evaluador:** Una manera (medida) de evaluar subconjuntos de atributos.

5.4.1.2.2 Búsqueda y Evaluación de los métodos de Selección de Atributos

Se van a enumerar, a continuación, los tipos de búsqueda y evaluación de los métodos de selección de atributos. Para más información se puede ampliar consultando en [64].

- **Ranker:**
  - **Búsqueda:** Ranker.
  - **Evaluador:**
    - **ChiSquareAttributeEval:** Usa el estadístico chi-cuadrado para evaluar el valor predictivo del atributo.
    - **GainRatio:** Usa gainratio.
    - **InfoGainAttributeEval:** Usa infogain.
- **Filter** y **Wrapper:**
  - **Búsqueda:**
    - **BestFirst:** Mejor primero (lento).
    - **ExhaustiveSearch:** Búsqueda exhaustiva (muy lento).
    - **GeneticSearch:** Búsqueda genética (rápida).
    - **GreedyStepWise:** Escalada (muy rápido).
    - **RankSearch:** Primero ordena los atributos y después construye el subconjunto de manera incremental, en dirección del mejor al peor, hasta que no merece la pena añadir nuevos atributos (rápido).
  - **Evaluador:**
    - **Filter:**
      - **CfsSubsetEval:** Evalúa el valor de un subconjunto de los atributos, considerando la capacidad individual de predicción de cada característica, junto con el grado de redundancia entre ellos.
    - **Wrapper:**
      - **ClassifierSubsetEval**: Evalúa subconjuntos de atributos en datos de entrenamiento o de test. Utiliza un clasificador para estimar el "mérito" de un conjunto de atributos.
      - **WrapperSubsetEval**: Evalúa subconjuntos de atributos usando aprendizaje. La validación cruzada se utiliza para estimar la exactitud de la actividad de aprendizaje para un subconjunto de atributos.

5.4.1.3 Selección de los clasificadores

A continuación, el usuario elegirá los clasificadores que quiera de entre los 12 siguientes. Para más información se puede ampliar consultando en [64].

- **NaiveBayes:** Clasificador bayesiano cuyos atributos numéricos son modelados como una distribución normal. Es un Clasificador discriminador bayes estándar. Para combinar las probabilidades de los distintos atributos asume que estas son independientes entre sí.

- **Logistic:** Crea y utiliza un modelo de regresión logística multinomial con un estimador. Hay algunas modificaciones del modelo propuesto por leCessie y van Houwelingen.

- **MultilayerPerceptron:** Redes neuronales en formas de perceptrón muliticapa entrenadas con el algoritmo backpropagation.

- **RBFNetwork**: Redes de funciones de base radial.

- **SMO:** Soluciona problemas multiclase usando clasificación por parejas. En los casos multi-clase las probabilidades predichas se juntaran usando los métodos de emparejamiento Hastie y Tibshirani.

- **VotedPerceptron:** Implementación del algoritmo de perceptrón votado de Freund y Schapire [75]. Reemplaza, globalmente, todos los valores que faltan, y transfroma los atributos nominales a binarios.

- **IB1:** Clasificador del vecino más próximo**.** Es una técnica perezosa, las cuales en vez de construir una estructura predictora (árbol de decisión, reglas,…), simplemente se guardan las instancias (los datos) o representantes de los mismos. Para clasificar un nuevo dato, simplemente se buscan las instancias más parecidas o cercanas.

- **IBk:** Clasificador de los k vecinos más próximos**.** Es una técnica perezosa.

- **KStar:** K* (K estrella) es un clasificador basado en instancias, donde la clase de una instancia de evaluación (test) se calcula usando una función de similitud con la clase de instancias de entrenamiento (train) parecidas. Se diferencia, de otros algoritmos de aprendizaje basados en instancias, en que utiliza una función de distancia basada en entropía. Es una técnica perezosa.

- **LWL:** Aprendizaje localmente ponderado. Utiliza un algoritmo basado en instancia para asignar pesos a las instancias que luego son utilizadas por un determinado manejador de pesos de instancias. Puede hacer clasificación, por ejemplo, usando Naive Bayes, o regresión, por ejemplo, mediante regresión lineal. Es una técnica perezosa.

- **J48:** Se trata de una implementación propia de Weka para el algoritmo C4.5, un algoritmo basado en clasificación por árbol de decisión.

- **REPTree:** Es un método de aprendizaje rápido mediante árboles de decisión. Construye un árbol de decisión usando la información de varianza y lo poda usando como criterio la

reducción del error. Solamente clasifica valores para atributos numéricos una vez. Los valores que faltan se obtienen partiendo las correspondientes instancias.

5.4.1.3.1 Técnicas de Entrenamiento

Para cada uno de estos clasificadores se puede elegir la técnica de entrenamiento, de entre dos posibles:

- **Percentage Split (Porcentaje de la división):** Esta técnica se indica por un porcentaje que dice la cantidad de instancias del fichero *arff* que se quiere usar como entrenamiento (train). Y la cantidad restante de instancias, será usada como evaluación o test.

- **Validación Cruzada:** Consiste en partir el conjunto de datos totales múltiples veces y calcular el porcentaje de aciertos medio. La idea es que los sesgos de unas y otras particiones se cancelen. Es posible, que por azar, los datos de entrenamiento y test estén sesgados, aunque estos hayan sido aleatorizados. Datos sesgados, significa que, por ejemplo, en los datos de entrenamiento aparezcan muchos más atributos a un cierto valor que en los datos de test, y el sistema creerá que hay una correlación entre este atributo y la clase. El valor que debe elegir el usuario, en la validación cruzada, es el número de celdas o trozos en las que se partirán los datos. Destacar que la Validación Cruzada siempre aleatorizará el fichero, pese a que el usuario haya elegido no Aleatorizarlo.

5.4.1.2 Escritura del fichero de salida

Finalmente el usuario terminará la creación del experimento y podrá guardarlo en un fichero *XML*, usando las librerías JDOM [65], con la estructura definida en la Figura 29. El fichero tiene una estructura jerárquica arbórea, donde Experimento es la raíz. Se observa una cabecera, con los atributos del *XML*. Después se encuentra la etiqueta del nodo padre, y las características de este conjunto (nombre, fecha en milisegundos, nº de arfss, el tipo de la selección de atributos (si tiene) y el número de clasificadores). Después se observa que primero se suceden los ficheros *arff*, los cuales tienen elementos que definen la información relativa a estas, como la ruta y el nombre para poder recuperarlas del disco duro. El tamaño, la fecha son atributos que no sirven para la carga posterior de los *arff*, pero que tiene carácter informativo. El número de atributos, el número de patrones, si se quiere aleatorizar, y si se quiere normalizar. Después vendrá la selección de atributos (si tiene), con la información de que tipo es, que búsqueda y que evaluador se usarán, y los parámetros de estos. Por último, se encuentran todos los clasificadores del experimento, con su nombre, su técnica de entrenamiento, y los parámetros de estos.

```
<?xml version="1.0" encoding="UTF-8"?>
<Experimento nombre="E" fecha="1283617948941" numero_arffs="2"
seleccion_atributos="Subconjuntos" numero_clasificadores="5">
  <Arff>
    <Nombre_Arff>ResultadosGif_aux.arff</Nombre_Arff>
    <Ruta>C:\Users\javier\Documents\PFC\PFC</Ruta>
    <Tamano>3528</Tamano>
    <Fecha>1283371051290</Fecha>
    <Numero_Atributos>57</Numero_Atributos>
    <Numero_Patrones>4</Numero_Patrones>
    <Aleatorizar>true</Aleatorizar>
    <Normalizar>true</Normalizar>
  </Arff>
.
.
  <Seleccion_Atributos>
    <Individual_Subconjuntos>Subconjuntos</Individual_Subconjuntos>
    <Filter_Wrapper>Wrapper</Filter_Wrapper>
    <Busqueda>BestFirst</Busqueda>
    <Evaluador>ClassifierSubsetEval</Evaluador>
    <Porcentaje_Ranker />
    <Clasificador_Evaluador>NaiveBayes</Clasificador_Evaluador>
    <Folds_Evaluador />
  </Seleccion_Atributos>
  <Clasificadores>
    <Nombre_Clasificador>NaiveBayes</Nombre_Clasificador>
    <Percentage_Cross>PercentageSplit</Percentage_Cross>
    <Percentage>66</Percentage>
    <Folds />
  </Clasificadores>
.
.
```

Cabecera e información del Experimento

Imágenes del Experimento

Selección de Atributos del Experimento

Clasificadores del Experimento

Figura 29: Ejemplo de fichero XML de un experimento

5.4.2 Ejecución de Experimentos (Train)

Un experimento está formado por uno o más ficheros de patrones *arff*, selección de atributos, o no, y uno o más clasificadores. La ejecución de experimentos, como su nombre indica, ejecutará experimentos de uno en uno, para que los clasificadores elegidos entrenen.

Figura 30: Proceso de ejecución de experimentos

Como se puede ver en la Figura 30, dado un experimento *XML* se ejecutará este, dando como salida los resultados en ficheros de texto TXT y los modelos, que son los clasificadores entrenados, en un fichero *model*. Habrá un modelo por cada uno de los clasificadores ejecutados. Si por ejemplo, hay dos ficheros *arff* en el experimento y tres clasificadores, se ejecutará cada uno de los clasificadores por cada *arff*, habiendo en total seis modelos. Los ficheros TXT de resultados tienen, en parte, el siguiente aspecto mostrado en la Figura 31:

```
=== Error on test split ===
Correctly Classified Instances         204               100      %
Incorrectly Classified Instances         0                 0      %
Kappa statistic                          1
Mean absolute error                      0
Root mean squared error                  0
Relative absolute error                  0      %
Root relative squared error              0      %
Total Number of Instances              204

=== Detailed Accuracy By Class ===
               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 1         0          1         1         1          ?        0.0
                 0         0          0         0         0          ?        1.0
Weighted Avg.    1         0          1         1         1          0

=== Confusion Matrix ===
   a   b   <-- classified as
 204   0 |   a = 0.0
   0   0 |   b = 1.0
```

Figura 31: Fichero TXT con los resultados de un experimento

Son una serie de valores, que dan información sobre los resultados de la ejecución. En este caso se muestran sólo los resultados de la parte de test, de unos datos divididos en 66% para entrenamiento y 33% para test. También habría unos resultados parecidos para la parte de entrenamiento. Se puede ver el número de instancias correctamente clasificadas, y su porcentaje, también el de incorrectamente clasificadas, varios tipos de errores, el porcentaje de TP (verdaderos positivos) y FP (falsos positivos). Estos valores se calculan usando la Confusion Matrix (matriz de confusión) que se encuentra en último lugar, donde se puede ver las clasificaciones hechas para cada clase. Los ficheros, *model*, de los modelos tienen un formato interno de Weka, que no es legible como fichero de texto.

5.4.3 Ejecución de Modelos (Test)

Una vez se han ejecutado los experimentos y se han obtenido los modelos (clasificadores entrenados), se pueden ejecutar estos modelos contra nuevos ficheros de instancias para evaluarlos y ver como clasifica las instancias, el modelo.

La ejecución de modelos sigue el siguiente diagrama, de la Figura 32:

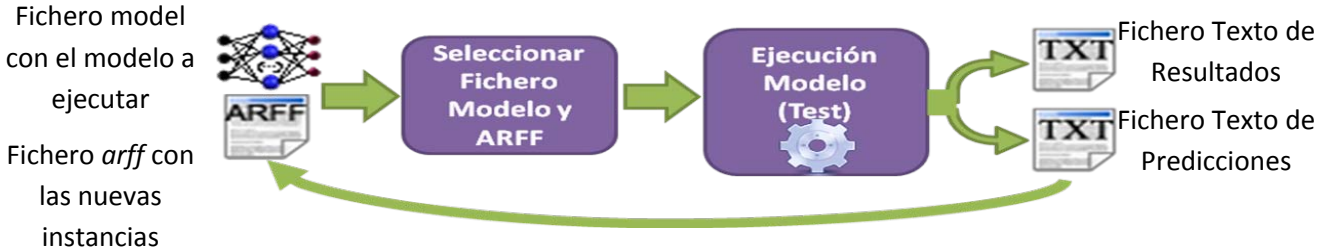

Figura 32: Diagrama de ejecución de modelos

Primero se cargarán de disco el fichero *model* con el modelo que se quiere evaluar, y el fichero *arff* con las nuevas instancias (no vistas antes por el modelo) que se quieren evaluar viendo como se clasifican. Destacar que, si el fichero *arff* con el que se entrenó al clasificador estaba normalizado, el fichero *arff* con las instancias de evaluación o test, deberán de normalizarse también. Después se ejecutará el modelo, y este dará dos salidas en ficheros de texto (TXT). Una es la predicción, que hace el modelo para cada instancia. Se indica en la Figura 33:

```
=== Predictions on test data ===

 inst#     actual  predicted error
prediction
     1      1:0.0      1:0.0       1
     2      1:0.0      1:0.0       1
     .
     .
```

Figura 33: Predicciones de la evaluación

La predicción indica, por cada instancia, el valor actual de la clase, el valor de la clase que se predice y el error. El valor actual, sólo sirve para compararlo con el valor predicho y calcular el error. El valor actual es el que tiene la instancia a evaluar. La evaluación de modelos se puede enfocar de dos formas distintas. Según el usuario haya decidido que instancias usar, el valor actual tendrá un significado u otro. Puede suceder que, el usuario haya asignado este valor arbitrariamente, en la extracción de medidas, sin saber realmente si tiene información oculta o no. Esto se hará así, cuando no se conozca la clase real de las instancias, ya que es lo que se quiere averiguar, con la predicción. En tal caso sólo es útil el valor predicho y no el error, ya que al usuario no le interesa saber cuánto se ha equivocado en la predicción, sino saber cuál es el valor predicho, ya que lo desconoce. En el caso contrario, el usuario conoce el valor real de la clase de cada instancia. Y lo que desea probar con la predicción es como de bien funciona el modelo. Y por lo tanto, el valor del error será importante en este caso. Este error se muestra en el fichero TXT de resultados, el cual, es bastante parecido al de resultados del entrenamiento, donde se especifica una tabla con los aciertos, errores, TP, FP, y matriz de confusión.

5.5 Implementación de un Estegoanalizador de Imágenes

El estegoanálisis es un proceso complejo de realizar, debido a la gran cantidad de factores que intervienen en el método de ocultación de información. Pero en este caso se ha desarrollado un estegoanalizador automático capaz de llevar a cabo este proceso. Para ello se realizaron una serie de experimentos eligiendo el mejor estegoanalizador resultante. Después este se implementó, en el propio programa, para que la herramienta pueda ser usada además para analizar imágenes.

Un estegoanalizador automático sirve para detectar información oculta en archivos, en este caso de imágenes, usando técnicas automáticas. Estas técnicas, derivadas del aprendizaje automático, funcionan en base a probabilidades, por lo que el hecho de encontrar una imagen, sospechosa de contener información oculta, no indica que, a ciencia cierta, esta imagen contenga información oculta. Sólo que esta imagen tiene características similares a las que tienen información oculta y por lo tanto es bastante probable que tenga información oculta.

Pero este tipo de razonamiento es suficientemente válido para detectar información oculta en imágenes de forma automática.

Figura 34: Diagrama de análisis de imágenes

Como se observa en la Figura 34, se parte de la búsqueda de imágenes de una ubicación del disco duro (o cualquier otro dispositivo con almacenamiento), pudiendo hacerse esta búsqueda de forma recursiva, para buscar en los subdirectorios de la ubicación. Esto puede ser útil si por ejemplo se quiere analizar un disco duro completo, habría que seleccionar como ubicación el directorio raíz del disco. Después se analizarán todas las imágenes del directorio (y subdirectorios) seleccionado. Para ello, primero se extraen las medidas que dieron los mejores resultados en los experimentos (medidas de la aleatoriedad), generando un fichero de instancias de las imágenes a analizar. A continuación, se ejecutará este fichero de instancias con el mejor modelo de la fase de experimentación. Finalmente se listarán sólo las imágenes que han dado resultado positivo en el análisis.

6. EXPERIMENTACIÓN

Dado que la herramienta ha sido ideada para la búsqueda de estegoanalizadores, se ha procedido a realizar una serie de experimentos para generar un estegoanalizador. De esta manera, se probará el funcionamiento general de la misma y su adecuación a la descripción funcional dada en el capítulo 4.

En primer lugar se hará una descripción general de los experimentos. A continuación se irán describiendo los diferentes módulos de ejecución: generación de conjuntos, extracción de medidas, entrenamiento y test.

6.1 Descripción de los Experimentos

Para la realización de los experimentos, en primer lugar se ha desarrollado un plan de cómo se van a desarrollar los experimentos. Debido al tiempo necesario para la ejecución de todos los experimentos se ha decidido realizar experimentos limitando las opciones con las que experimentar.

Se ha partido de un banco de 2000 imágenes en formato JPG. Han sido descargadas de internet con la herramienta NeoDownloader [76], en su versión de prueba, la cual permite descargas masivas de imágenes de una página web, de forma recursiva. Se usaron varias webs para obtener este banco de imágenes [77, 78, 79]. Del conjunto total de imágenes, se han creado dos subconjuntos: uno con el 70% de las imágenes y otro con el 30%, que se utilizarán para el entrenamiento y test de los algoritmos de IA. Se parte de la hipótesis de que las imágenes no tienen información oculta, pero no se puede tener certeza absoluta. Para tener algo más de fiabilidad en este aspecto se han utilizado StegDetect [57] y StegSecret [61].

Para ocultar información en imágenes se ha utilizado únicamente Vecna [38]. Hay varias razones por las que sólo se usa esta herramienta. Una razón es el tiempo, ya que por cada herramienta a analizar hay que repetir todo el experimento. Y no sólo se consume tiempo por ocultar información, sino que la extracción de medidas lleva bastante tiempo, si se tiene en cuenta que se extraen cuatro ficheros de medidas por cada conjunto de imágenes. Además, Vecna es una herramienta recientemente realizada, por lo que no se han hecho estudios sobre ella.

La ocultación se ha realizado en las mismas 2000 imágenes de partida. De esta forma se obtienen así dos conjuntos: 2000 imágenes sin información oculta y 2000 imágenes con información oculta. Si bien el procedimiento ideal requeriría utilizar diferentes imágenes para la ocultación, esto no es tan problemático ya que los algoritmos de IA no saben relacionar las parejas de imágenes, por lo que las tratan como instancias de imágenes diferentes.

El esquema seguido para realizar los experimentos en la herramienta, se resume en la Figura 35:

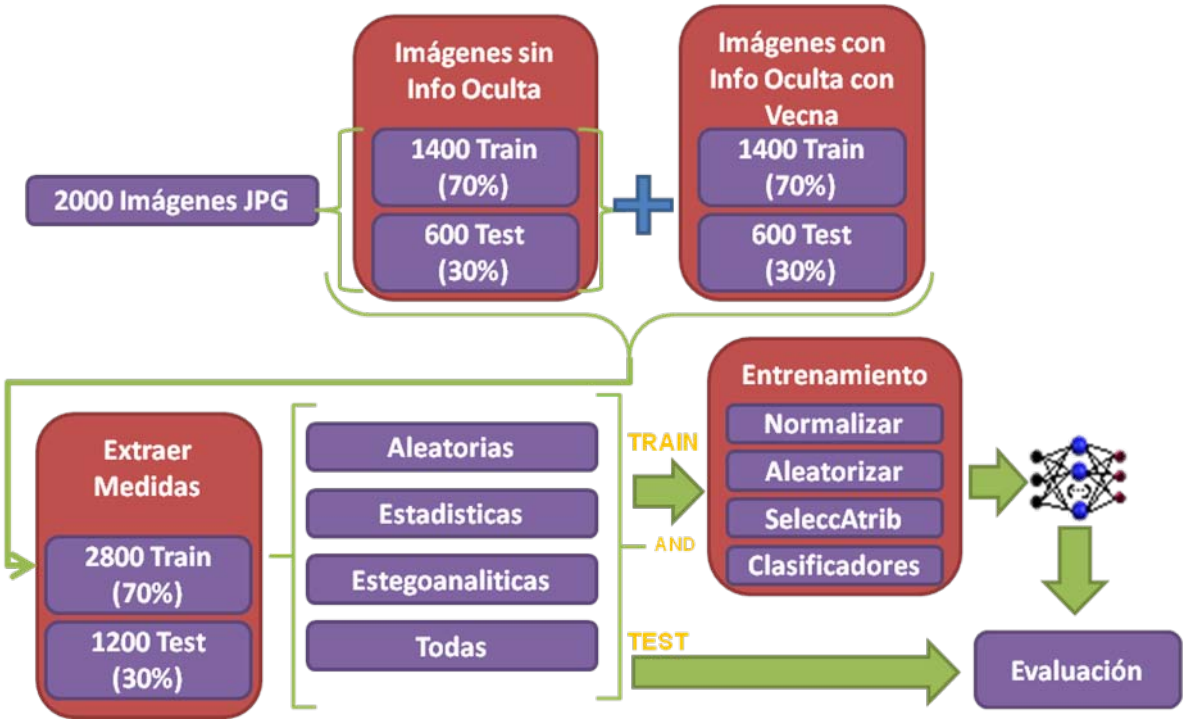

Figura 35: Diagrama del plan de la experimentación

Aunque la experimentación se ha realizado con una sola herramienta esteganográfica, JUBSAC permite la utilización de todas las herramientas por separado e incluso todas a la vez. Para ello habría que crear bancos de imágenes en todos los formatos, ya que no todas las herramientas funcionan con los mismos formatos. Para esto usar la herramienta de *Conversión de Imágenes* disponible en JUBSAC. Y finalmente repetir el diagrama anterior para cada herramienta. Para la ejecución de los experimentos se ha usado el PC de sobremesa para la experimentación de JUBSAC, cuyas características se detallan en el capítulo 8 (Gestión del Proyecto), en la sección 8.2 (Medios Empleados).

6.2 Generación de Conjuntos

Tal como se ha explicado en la sección 6.1, se han creado dos conjuntos de imágenes iniciales:

- TrainJPG_Conjunto.xml: conjunto para las 1400 imágenes en formato JPG de Train.
- TestJPG_Conjunto.xml: conjunto para las 600 imágenes en formato JPG de Test.

Para la generación de los conjuntos con información oculta se ha ejecutado Vecna sobre los dos conjuntos (mediante JUBSAC). El tiempo de cada ejecución se resume en la Tabla 11:

TAREA	ENTRADA	SALIDA	TIEMPO
Ejecución Vecna	TrainJPG_Conjunto.xml	Train_JPG_Conjunto_steg_Vecna.xml	1d,3h,49m,24s
Ejecución Vecna	TestJPG_Conjunto.xml	Train_JPG_Conjunto_steg_Vecna.xml	9h,45m,4s

Tabla 11: Tiempos de la ejecución de Vecna

Cabe destacar que el tiempo es bastante elevado, usando sólo una única herramienta, que además no es de las más lentas. Además, la ocultación de información se hace de forma paralela, usando los dos núcleos de procesamiento de los que se dispone.

6.3 Extracción de Medidas

Una vez se han generado los conjuntos de imágenes, se ha procedido a la extracción de características de las imágenes, de los cuatro conjuntos de los que se dispone. Estas características serán utilizadas durante el entrenamiento, para que los clasificadores aprendan, y durante el test para validar los modelos.

Tal como se muestra en la sección 5.3, se han extraído 4 tipos de medidas:

- **Aleatoriedad**: sólo se han utilizado las medidas de aleatoriedad y el formato de la imagen (**99 atributos** en total). Para las porciones de los bytes a tomar se tomó sólo un bit, el bit menos significativo.
- **Estadísticas**: sólo se han utilizado las medidas estadísticas y el formato de la imagen (**71 atributos** en total).
- **Estegoanalíticas**: sólo se han utilizado las medidas estegoanalíticas, menos la tasa de las parejas de color debido a su alto coste temporal, y el formato de la imagen (**23 atributos** en total).
- **Todas las medidas**: se han utilizado todas las medidas anteriores (**191 atributos** en total).

Se han extraído estos tipos de medidas para poder comprobar con qué tipo de medidas se genera el mejor clasificador. Se ha usado también el formato de la imagen, ya que se consideró, en principio, importante que la herramienta pudiera correlacionar algún atributo

con el formato. Pudiendo así darse cuenta de que las herramientas esteganográficas usan más unos formatos que otros. El formato se añadió por estas razones, aunque finalmente sólo se hayan realizado experimentos para un único formato (JPG), por problemas de recursos. De todas formas, los clasificadores obviarán este atributo como determinante para calcular la clase.

En la Tabla 12 se muestran los tiempos de la extracción de medidas de cada tipo, para las imágenes sin información oculta.

TAREA	ENTRADA	SALIDA	TIEMPO
Extraer Medidas de Aleatoriedad	TrainJPG_Conjunto.xml	TrainJPG_Conjunto_Aleatoriedad.*arff*	3h,48m,55s
Extraer Medidas de Aleatoriedad	TestJPG_Conjunto.xml	TestJPG_Conjunto_Aleatoriedad.*arff*	1h,15m,26s
Extraer Medidas Estadísticas	TrainJPG_Conjunto.xml	TrainJPG_Conjunto_Estadisticas.*arff*	19h,33s
Extraer Medidas Estadísticas	TestJPG_Conjunto.xml	TestJPG_Conjunto_Estadisticas.*arff*	5h,1m,16s
Extraer Medidas Estegoanalíticas	TrainJPG_Conjunto.xml	TrainJPG_Conjunto_Estegoanaliticas.*arff*	7h,31m,4s
Extraer Medidas Estegoanalíticas	TestJPG_Conjunto.xml	TestJPG_Conjunto_Estegoanaliticas.*arff*	2h,36m,46s
Extraer Todas las Medidas	TrainJPG_Conjunto.xml	TrainJPG_Conjunto_Todas.*arff*	1d,1h,34m,19s
Extraer Todas las Medidas	TestJPG_Conjunto.xml	TestJPG_Conjunto_Todas.*arff*	9h,2m,19s

Tabla 12: Tiempos de la extracción de medidas de conjuntos sin información oculta

Como se observa la extracción de medidas consume una gran cantidad de tiempo. Esto es debido a que hay 196 medidas en total, y por cada medida se deben recorrer todos los píxeles de cada imagen. Pese a que la extracción de todas las medidas consumía mucho tiempo, se procedió a realizar esta de forma automática con la herramienta, para así ponerla a prueba, extrayendo una gran cantidad de medidas, durante bastante tiempo y así confirmar la robustez de la herramienta.

En la Tabla 13 se muestran los tiempos de la extracción de medidas de cada tipo para las imágenes con información oculta, habiendo usado Vecna.

TAREA	ENTRADA	SALIDA	TIEMPO
Extraer Medidas de Aleatoriedad	Train_JPG_Conjunto_steg_Vecna.xml	Train_JPG_Conjunto_steg_Vecna _Aleatoriedad.*arff*	1h,5m,24m,19s
Extraer Medidas de Aleatoriedad	Test_JPG_Conjunto_steg_Vecna.xml	Test_JPG_Conjunto_steg_Vecna _Aleatoriedad.*arff*	9h,50m,59s
Extraer Medidas Estadísticas	Train_JPG_Conjunto_steg_Vecna.xml	Train_JPG_Conjunto_steg_Vecna _Estadisticas.*arff*	14h,59m,35s
Extraer Medidas Estadísticas	Test_JPG_Conjunto_steg_Vecna.xml	Test_JPG_Conjunto_steg_Vecna _ Estadisticas.*arff*	5h,11m,56s
Extraer Medidas Estegoanalíticas	Train_JPG_Conjunto_steg_Vecna.xml	Train_JPG_Conjunto_steg_Vecna _Estegoanaliticas.*arff*	7h,54m,33s
Extraer Medidas Estegoanalíticas	Test_JPG_Conjunto_steg_Vecna.xml	Test_JPG_Conjunto_steg_Vecna _ Estegoanaliticas.*arff*	2h,47m,45s
Extraer Todas las Medidas	Train_JPG_Conjunto_steg_Vecna.xml	Train_JPG_Conjunto_steg_Vecna _Todas.*arff*	2d,6h,17m,11s
Extraer Todas las Medidas	Test_JPG_Conjunto_steg_Vecna.xml	Test_JPG_Conjunto_steg_Vecna _ Todas.*arff*	18h,5s

Tabla 13: Tiempos de la extracción de medidas de conjuntos con información oculta con Vecna

A continuación, para la ejecución de los algoritmos de IA, se fusionaron los conjuntos de datos de imágenes con información oculta y sin ella. Esto se hizo, por cada uno de los cuatro tipos de ficheros *arff* (medidas de aleatoriedad, estadísticas, estegoanalíticas y todas). De estos cuatro tipos de ficheros se obtuvo un solo fichero, para cada uno de los cuatro tipos. Se realizó tanto para los ficheros de train como para los de test. Teniendo en total ocho ficheros *arff*. Al fusionarlos se obtiene que para los ficheros de patrones de train existen 2800

instancias, y para los de test 1200. Resultado de juntar 1400 instancias sin información oculta, con 1400 con información oculta, para train. Y por otro lado de juntar 600 instancias sin información oculta y otras 600 con información oculta, para test. Para entender mejor este último paso se representa el proceso de fusionado, sólo, para los ficheros con las medidas aleatorias, en la Figura 36. Este proceso se repite para los cuatro tipos de medidas.

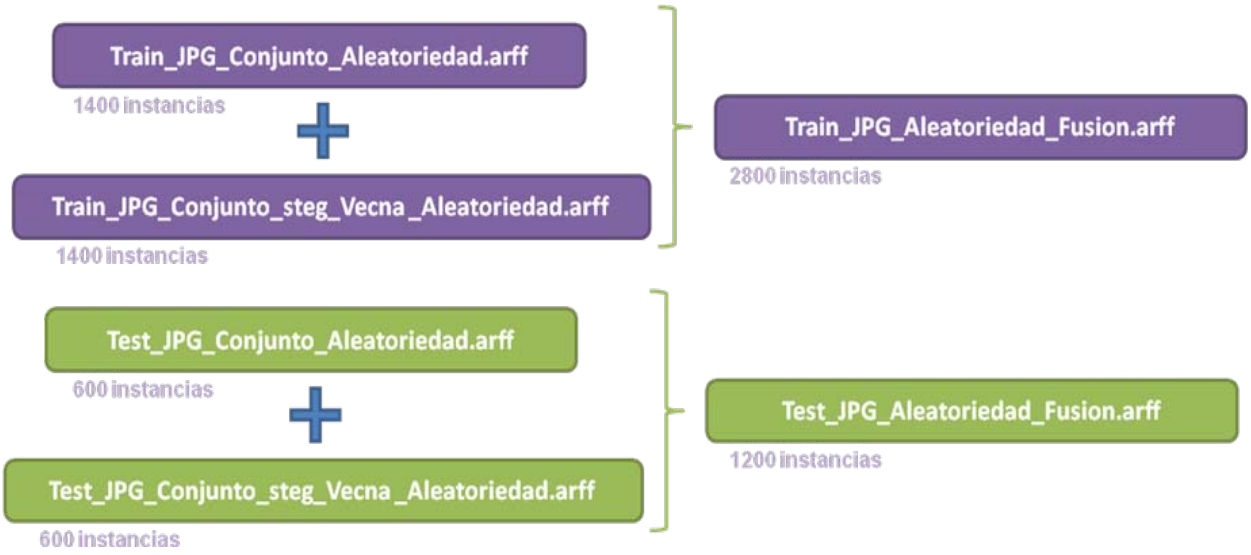

Figura 36: Diagrama del fusionado de ficheros arff de train y test

Para los ficheros de entrenamiento (train) hay 2400 instancias: 1400 con la clase 0, ya que las imágenes de las que se calcularon no tenían información oculta y otras 1400 con la clase 1, ya que las imágenes de las que se calcularon tenían información oculta. Al tener el mismo número de instancias para cada clase, se dice que las clases están equilibradas. Se ha decidido hacer así para dar el mismo peso a los dos tipos de instancias, aunque en el mundo real, los datos no se suelen poder recolectar en proporciones iguales para todas las clases. Si las clases estuvieran desequilibradas, es decir, tuvieran distintos número de instancias para cada una, los clasificadores darían más peso a las instancias con la clase mayoritaria, y por lo tanto no aprendería de la mejor forma posible a generalizar. Esto ocurre en los casos en los que hay una gran diferencia entre el número de instancias de una clase y otra. Los clasificadores aprenden de ejemplos, y siempre es necesario intentar equilibrar al máximo el número de ejemplos de cada una de las clases.

6.4 Ejecución de Experimentos para Entrenamiento

El siguiente paso es realizar experimentos para entrenar los cuatro ficheros de entrenamiento con los clasificadores disponibles.

6.4.1 Preprocesado de los datos

Con la herramienta de creación de experimentos de JUBSAC, se crearon cuatro experimentos. Uno por cada fichero *arff* de entrenamiento. En todos los experimentos se ha realizado un proceso de selección de atributos, debido al gran número de atributos (i.e. 191 en el caso de todas las medidas).

En instancias con una gran cantidad de atributos, podría haber muchos atributos irrelevantes, para el aprendizaje del clasificador y sería recomendable hacer un filtrado de estos para quedarse con los mejores. Para no gastar excesiva cantidad de tiempo y que aún así el método fuera eficaz, el tipo de selección que se utilizó es individual, es decir, se usó un *Ranker* para la búsqueda y para la evaluación de atributos se usó el evaluador estadístico chi-cuadrado. Se decidió que *Ranker* seleccionara el 50% de los atributos ordenados de mejor a peor, en cuanto a correlación de estos con la clase. Esta selección se hizo para todos los experimentos menos para las de medidas estegoanalíticas, ya que el número total de atributos es de 23, y reducir este puede provocar que el clasificador no llegue a aprender correctamente, así que el *Ranker* seleccionó el 100% de los atributos.

Se decidió aleatorizar las instancias, para desordenar estas y eliminar posibles sesgos, ya que puede haber instancias seguidas que tengan ciertos atributos idénticos, como la clase. También se normalizaron los atributos de las instancias para que no hubiera una gran diferencia numérica entre estos, y los clasificadores pudieran dar más peso a unos atributos frente a otros.

Para entrenar se usaron los 12 clasificadores disponibles, dividiendo las 2800 instancias de entrenamiento en 66% (1848 instancias) para entrenamiento de los clasificadores y 33% (952 instancias) de evaluación de estos, por parte de Weka. Los clasificadores se ejecutaban en parejas de forma paralela.

En las tablas que se muestran a continuación, se analizarán los siguientes atributos del entrenamiento:

- **Tiempo Entrenamiento**: Es el tiempo de cada clasificador, en acabar el entrenamiento.
- **Porcentaje de Aciertos**: Es el porcentaje de aciertos en las instancias de test (33%) para el entrenamiento.
- **Error Absoluto**: Es el error absoluto medio dividido por el correspondiente error del clasificador.
- **Media de falsos positivos**: Es el número de falsos positivos entre las dos clases, para el test (33%) del entrenamiento.

6.4.2 Entrenamiento con Medidas de Aleatoriedad

A continuación, en la Tabla 14, se muestran los tiempos de ejecución, del entrenamiento, de los clasificadores con la selección de atributos para los ficheros de instancias con las medidas de aleatoriedad. Y los resultados de la evaluación con el 33% de las 2800 instancias (952 instancias).

CLASIFICADOR	TIEMPO ENTRENAMIENTO	% ACIERTOS	ERROR ABSOLUTO	MEDIA FALSOS POSITIVOS
NaiveBayes	11s	100%	0%	0
Logistic	8s	100%	0%	0
MultilayerPerceptron	3m,14s	100%	0.0989%	0
RBFNetwork	1m,31s	100%	0%	0
SMO	1m,3s	100%	0%	0
VotedPerceptron	1m,53s	100%	0%	0
IB1	54s	100%	0%	0
IBk	55s	100%	0.108%	0
KStar	13m,32s	99.0801%	0.9204%	0.01
LWL	3m,55s	100%	0%	0
J48	5s	100%	0%	0
REPTree	5s	100%	0%	0

Tabla 14: Resultados del Entrenamiento con Medidas Aleatorias

La selección de atributos previa al entrenamiento, con los atributos de las instancias normalizadas, da una puntuación a todos los atributos (en función de su correlación con la clase) y los ordena en función de esta puntuación. Después selecciona solamente la primera mitad de estos, para finalmente entrenar con ellos.

En la Figura 37, se muestra el 50% con los mejores atributos (49 atributos de un total de 99) y su puntuación son:

```
Ranked attributes:
1848          83 ErrorMonteCarloCanalesYPorcionesRG
1848          44 CompresionCanalesYPorcionesG
1848          82 MonteCarloCanalesYPorcionesRG
1848          86 CompresionCanalesYPorcionesRB
1848          41 ErrorMonteCarloCanalesYPorcionesR
1848          40 MonteCarloCanalesYPorcionesR
1848          51 CompresionCanalesYPorcionesB
1848          55 ErrorMonteCarloCanalesYPorcionesB
1848          54 MonteCarloCanalesYPorcionesB
1848          79 CompresionCanalesYPorcionesRG
1848          48 ErrorMonteCarloCanalesYPorcionesG
1848          47 MonteCarloCanalesYPorcionesG
1848          33 MonteCarloPorciones
1848          93 CompresionCanalesYPorcionesGB
1848          30 CompresionPorciones
1848          96 MonteCarloCanalesYPorcionesGB
1848          97 ErrorMonteCarloCanalesYPorcionesGB
1848          90 ErrorMonteCarloCanalesYPorcionesRB
1848          37 CompresionCanalesYPorcionesR
1848          89 MonteCarloCanalesYPorcionesRB
1848          34 ErrorMonteCarloPorciones
1848          53 MediaCanalesYPorcionesB
1848          29 EntropiaPorciones
1848          39 MediaCanalesYPorcionesR
……
```

```
……
1848          46 MediaCanalesYPorcionesG
1848          36 EntropiaCanalesYPorcionesR
1848          50 EntropiaCanalesYPorcionesB
1848          32 MediaPorciones
1848          43 EntropiaCanalesYPorcionesG
1848          78 EntropiaCanalesYPorcionesRG
1848          88 MediaCanalesYPorcionesRB
1848          85 EntropiaCanalesYPorcionesRB
1848          81 MediaCanalesYPorcionesRG
1848          95 MediaCanalesYPorcionesGB
1848          99 Formato
1848          92 EntropiaCanalesYPorcionesGB
1844.00236    42 CorrelacionCanalesYPorcionesR
1844.00236    56 CorrelacionCanalesYPorcionesB
1844.00236    91 CorrelacionCanalesYPorcionesRB
1829.29196    87 ChiCanalesYPorcionesRB
1791.52903    38 ChiCanalesYPorcionesR
1791.2324     52 ChiCanalesYPorcionesB
1676.16771    35 CorrelacionPorciones
1671.89196    84 CorrelacionCanalesYPorcionesRG
1671.35159    98 CorrelacionCanalesYPorcionesGB
1662.30541    49 CorrelacionCanalesYPorcionesG
1649.32928    22 EntropiaCanalesB
1614.14895    80 ChiCanalesYPorcionesRG
1604.80806    94 ChiCanalesYPorcionesGB
```

Figura 37: Selección Atributos en las Medidas Aleatorias

Se observa en la Tabla 14 que los resultados son excelentes. En los atributos seleccionados por Ranker, se observa que la mayoría son atributos relacionados con medidas de componentes RGB en porciones de un bit. Y es con estos con los se obtienen estos excelentes resultados.

6.4.3 Entrenamiento con Medidas Estadísticas

A continuación, en la Tabla 15, se muestran los tiempos de ejecución, del entrenamiento, de los clasificadores con la selección de atributos para los ficheros de instancias con las medidas estadísticas. Y los resultados de la evaluación con el 33% de las 2800 instancias (952 instancias).

CLASIFICADOR	TIEMPO ENTRENAMIENTO	% ACIERTOS	ERROR ABSOLUTO	MEDIA FALSOS POSITIVOS
NaiveBayes	8s	99.4748%	1.9296%	0.006
Logistic	7s	100%	0%	0
MultilayerPerceptron	1m,26s	100%	0.1568%	0
RBFNetwork	2m,33s	99.3697%	2.0148%	0.007
SMO	1m,7s	100%	0%	0
VotedPerceptron	1m,6s	100%	0%	0
IB1	1m,5s	100%	0%	0
IBk	1m,5s	100%	0.108%	0

KStar	3m,46s	92.0168%	31.2905%	0.079
LWL	2m,9s	100%	0%	0
J48	4s	100%	0%	0
REPTree	4s	100%	0%	0

Tabla 15: Resultados del entrenamiento con medidas estadísticas

La selección de atributos previa al entrenamiento da el 50% con los mejores atributos (35 atributos de un total de 71) y su puntuación se muestra en la Figura 40:

```
Ranked attributes:
1848        71 Formato
   0        24 CVPorciones
   0        23 DesviacionPorciones
   0        26 MediaPixelesCanalesYPorcionesR
   0        25 CurtosisPorciones
   0        20 CurtosisCanalesB
   0        19 CVCanalesB
   0        22 VarianzaPorciones
   0        21 MediaPixelesPorciones
   0        33 DesviacionCanalesYPorcionesG
   0        32 VarianzaCanalesYPorcionesG
   0        35 CurtosisCanalesYPorcionesG
   0        34 CVCanalesYPorcionesG
   0        28 DesviacionCanalesYPorcionesR
   0        27 VarianzaCanalesYPorcionesR
   0        31 MediaPixelesCanalesYPorcionesG
   0        29 CVCanalesYPorcionesR
   0        30 CurtosisCanalesYPorcionesR
……
```

```
……
   0         7 VarianzaCanalesR
   0         6 MediaPixelesCanalesR
   0         9 CVCanalesR
   0         8 DesviacionCanalesR
   0         2 VarianzaImagen
   0         1 MediaPixelesImagen
   0         5 CurtosisImagen
   0         3 DesviacionImagen
   0         4 CVImagen
   0        16 MediaPixelesCanalesB
   0        15 CurtosisCanalesG
   0        18 DesviacionCanalesB
   0        17 VarianzaCanalesB
   0        11 MediaPixelesCanalesG
   0        10 CurtosisCanalesR
   0        14 CVCanalesG
   0        12 VarianzaCanalesG
```

Figura 38: Selección de Atributos en las medidas estadísticas

Se observa, en la Figura 38, que la puntuación que da Ranker a los atributos es 0, menos para el formato. Lo que significa que el evaluador usado chi-cuadrado no encuentra una correlación importante entre los atributos estadísticos y la clase. Aún así, la selección de atributos se evalúa de forma individual, midiendo la correlación de cada atributo con la clase, por lo que no detecta si hay conjuntos de atributos que si influyen en la clase. Pero esto no es tan importante, ya que el clasificador obtiene unos buenos resultados en el 33% de la evaluación. Aún así se observa, en la Tabla 15, que los resultados no son tan buenos como con las medidas aleatorias, ya que ahora hay tres clasificadores que bajan del 100%, y uno de ellos, el KStar, está más cerca del 90% que del 100%, y tiene una media mayor de falsos positivos. Aunque este clasificador también era el peor para las medidas aleatorias.

6.4.4 Entrenamiento con Medidas Estegoanalíticas

A continuación, en la Tabla 16, se muestran los tiempos de ejecución, del entrenamiento, de los clasificadores con la selección de atributos para los ficheros de instancias con las medidas estegoanalíticas. Y los resultados de la evaluación con el 33% de las 2800 instancias (952 instancias).

CLASIFICADOR	TIEMPO ENTRENAMIENTO	% ACIERTOS	ERROR ABSOLUTO	MEDIA FALSOS POSITIVOS
NaiveBayes	6s	82.9832%	33.5079%	0.161
Logistic	6s	99.7899%	0.4196%	0.002
MultilayerPerceptron	1m,7s	100%	0.1676%	0
RBFNetwork	54s	100%	0%	0
SMO	33s	100%	0%	0
VotedPerceptron	33s	100%	0%	0
IB1	32s	99.895%	0.2099%	0.001
IBk	33s	99.895%	0.3176%	0.001
KStar	10m,3s	85.7143%	30.1834%	0.147
LWL	2m,14s	100%	0%	0
J48	4s	100%	0%	0
REPTree	4s	100%	0%	0

Tabla 16: Resultados del entrenamiento con medidas estegoanalíticas

En este experimento, debido al escaso número de atributos de las instancias (23 atributos en total), la selección se hizo de todos los atributos al 100%, cuyo resultado son los mismos atributos pero ordenados en función de la puntuación que les da Ranker. Estos atributos ordenados son los mostrados en la Figura 39:

```
Ranked attributes:
 1848        23 Formato
 560.513    18 LongitudVerdeSPA
 549.2583    4 LongitudVerdeRSANoSuperponerGrupos
 534.0256   10 LongitudVerdeRSASuperponerGrupos
 517.9051   17 PorcentajeVerdeSPA
 497.9716    3 PorcentajeVerdeRSANoSuperponerGrupos
 482.4686    9 PorcentajeVerdeRSASuperponerGrupos
 441.9271   16 LongitudRojoSPA
 412.7223    2 LongitudRojoRSANoSuperponerGrupos
 397.5662    8 LongitudRojoRSASuperponerGrupos
 395.1459   15 PorcentajeRojoSPA
 384.5523   22 MediaLongitudesSPA
 372.8065    7 PorcentajeRojoRSASuperponerGrupos
 346.2329   21 MediaResultadosSPA
 340.7003    1 PorcentajeRojoRSANoSuperponerGrupos
 331.8577   14 MediaLongitudesRSA
 295.1368   13 MediaResultadosRSA
 284.1862   20 LongitudAzulSPA
 271.5182    6 LongitudAzulRSANoSuperponerGrupos
 270.3982   12 LongitudAzulRSASuperponerGrupos
 249.502    19 PorcentajeAzulSPA
 238.5352    5 PorcentajeAzulRSANoSuperponerGrupos
 233.0991   11 PorcentajeAzulRSASuperponerGrupos
```

Figura 39: Selección de atributos en las medidas estegoanalíticas

Se observa en la Tabla 16 que los resultados son peores que los obtenidos con las medidas aleatorias y estadísticas. Se puede apreciar que algunos aciertos rondan el 80% de aciertos, por lo que los fallos son del 20% y esto puede ser demasiado error para el estegoanalizador. Debido a estos resultados se supone que en la evaluación se obtendrán bastantes más fallos, con estas medidas que con las anteriores. Aunque no todos los modelos clasifican mal con las medidas estegoanalíticas.

6.4.5 Entrenamiento con Todas las Medidas

A continuación, en la Tabla 17, se muestran los tiempos de ejecución, del entrenamiento, de los clasificadores con la selección de atributos para los ficheros de instancias con todas las medidas anteriores. Y los resultados de la evaluación con el 33% de las 2800 instancias (952 instancias).

CLASIFICADOR	TIEMPO ENTRENAMIENTO	% ACIERTOS	ERROR ABSOLUTO	MEDIA FALSOS POSITIVOS
NaiveBayes	18s	100%	0%	0
Logistic	13s	100%	0%	0
MultilayerPerceptron	9m,28s	100%	0.0708%	0
RBFNetwork	2m,32s	100%	0%	0
SMO	2m,4s	100%	0%	0
VotedPerceptron	2m,2s	100%	0%	0
IB1	2m,2s	100%	0%	0
IBk	2m,2s	100%	0.108 %	0
KStar	30m,50s	99.0546%	0.9444%	0.009
LWL	8m,20s	100%	0%	0
J48	9s	100%	0%	0
REPTree	8s	100%	0%	0

Tabla 17: Resultados del entrenamiento con todas las medidas

La selección de atributos, previa al entrenamiento, da el 50% con los mejores atributos (95 atributos de un total de 191) y las primeras puntuaciones se muestran en la Figura 40:

```
Ranked attributes:
1848           83 ErrorMonteCarloCanalesYPorcionesRG
1848           86 CompresionCanalesYPorcionesRB
1848           55 ErrorMonteCarloCanalesYPorcionesB
1848           82 MonteCarloCanalesYPorcionesRG
1848           79 CompresionCanalesYPorcionesRG
1848           96 MonteCarloCanalesYPorcionesGB
1848           97 ErrorMonteCarloCanalesYPorcionesGB
1848           89 MonteCarloCanalesYPorcionesRB
1848           93 CompresionCanalesYPorcionesGB
1848           90 ErrorMonteCarloCanalesYPorcionesRB
1848           37 CompresionCanalesYPorcionesR
1848           40 MonteCarloCanalesYPorcionesR
1848           30 CompresionPorciones
1848           34 ErrorMonteCarloPorciones
1848           33 MonteCarloPorciones
1848           48 ErrorMonteCarloCanalesYPorcionesG
1848           54 MonteCarloCanalesYPorcionesB
1848           51 CompresionCanalesYPorcionesB
1848           41 ErrorMonteCarloCanalesYPorcionesR
1848           47 MonteCarloCanalesYPorcionesG
1848           44 CompresionCanalesYPorcionesG
1848           81 MediaCanalesYPorcionesRG
1848           53 MediaCanalesYPorcionesB
1848           78 EntropiaCanalesYPorcionesRG
……
```

```
……
1848           92 EntropiaCanalesYPorcionesGB
1848           95 MediaCanalesYPorcionesGB
1848           85 EntropiaCanalesYPorcionesRB
1848           88 MediaCanalesYPorcionesRB
1848           32 MediaPorciones
1848           36 EntropiaCanalesYPorcionesR
1848           29 EntropiaPorciones
1848          191 Formato
1848           46 MediaCanalesYPorcionesG
1848           50 EntropiaCanalesYPorcionesB
1848           39 MediaCanalesYPorcionesR
1848           43 EntropiaCanalesYPorcionesG
1844.00236     42 CorrelacionCanalesYPorcionesR
1844.00236     91 CorrelacionCanalesYPorcionesRB
1844.00236     56 CorrelacionCanalesYPorcionesB
1829.29196     87 ChiCanalesYPorcionesRB
1791.52903     38 ChiCanalesYPorcionesR
1791.2324      52 ChiCanalesYPorcionesB
1676.16771     35 CorrelacionPorciones
1668.60907     84 CorrelacionCanalesYPorcionesRG
1668.05188     98 CorrelacionCanalesYPorcionesGB
1662.30541     49 CorrelacionCanalesYPorcionesG
1649.32928     22 EntropiaCanalesB
1614.14895     80 ChiCanalesYPorcionesRG
1604.80806     94 ChiCanalesYPorcionesGB
```

Figura 40: Selección de atributos con todas las medidas

Se puede contemplar en la Tabla 17 que los resultados son excelentes. Igual de buenos que para las medidas aleatorias, y esto es debido a que los mejores atributos seleccionados son en su mayor parte medidas aleatorias. Esto hace ver que para la elección del mejor modelo será más recomendable calcular sólo las medidas aleatorias, en lugar de todas. Así, para cuando se construya el estegoanalizador en JUBSAC, este tardará menos, aumentando de esta forma la eficiencia. Sólo se tendrán que calcular 99 medidas aleatorias y no 191. Y además de aumentar la eficiencia no disminuirá la eficacia, al calcular menos medidas, ya que los resultados son similares entre las medidas aleatorias y todas las medidas.

Viendo todas las tablas anteriores se puede comprobar que el clasificador KStar no es recomendable, no sólo debido a que sus aciertos fluctúan bastante, sino que consume mucho tiempo, comparando con los demás clasificadores.

6.5 Evaluación de Experimentos para Test

A continuación, se muestra el resultado de la predicción al ejecutar los clasificadores entrenados previamente con los cuatro tipos de ficheros de instancias, según las medidas. Los modelos se han ejecutado con 1200 instancias nuevas, donde 600 no tienen información oculta y las otras 600 sí. Ya que los datos de entrenamiento fueron normalizados, estos también, porque el clasificador aprendió con datos normalizados. Estas 1200 instancias son el 30% de los conjuntos iniciales de los que se partía para realizar el entrenamiento, 4000 imágenes (2000 sin información oculta y 2000 con ella).

Destacar que, en todas las evaluaciones aquí detalladas, el modelo de IBk ejecuta en clasificación con k=1 (nº de vecinos más próximos), ya que así se ha construido el modelo durante el entrenamiento. Esto implica que los resultados de este serán idénticos a IB1, teniendo así 11 clasificadores reales en vez de 12.

En las tablas que se muestran a continuación, se analizarán los siguientes atributos de la evaluación:

- **Tiempo Ejecución**: Es el tiempo que tarda del modelo, en evaluar todas las instancias.
- **Porcentaje de Aciertos**: Es el porcentaje de aciertos en las instancias que se están evaluando.
- **Instancias Correctamente Clasificadas**: Es el número de instancias que el modelo clasifica de forma acertada.
- **Instancias Incorrectamente Clasificadas**: Es el número de instancias que el modelo clasifica de forma errónea.
- **Error Absoluto Medio**: Es el valor absoluto de la diferencia, media, entre las instancias originales y las predichas.
- **Media de falsos positivos**: Es el número de falsos positivos entre las dos clases, para la evaluación.

6.5.1 Evaluación Modelos de Medidas de Aleatoriedad

En la Tabla 18 se muestran los resultados de la evaluación de los modelos entrenados con medidas de aleatoriedad:

CLASIFICADOR	TIEMPO EJECUCIÓN	% ACIERTOS	INSTANCIAS CORRECTAMENTE CLASIFICADAS	INSTANCIAS INCORRECTAMENTE CLASIFICADAS	ERROR ABSOLUTO MEDIO	MEDIA FALSOS POSITIVOS
NaiveBayes	2s	100%	1200	0	0	0
Logistic	2s	100%	1200	0	0	0
MultilayerPerceptron	3s	100%	1200	0	0.0004	0
RBFNetwork	2s	100%	1200	0	0	0
SMO	2s	100%	1200	0	0	0
VotedPerceptron	2s	100%	1200	0	0	0
IB1	30s	100%	1200	0	0	0
IBk	17s	100%	1200	0	0.0005	0
KStar	19m,52s	100%	1200	0	0	0
LWL	6m,56s	99.9167%	1199	1	0.0008	0.001
J48	2s	100%	1200	0	0	0
REPTree	2s	100%	1200	0	0	0

Tabla 18: Resultados de la evaluación de modelos con medidas aleatorias

Se puede observar que los resultados de evaluar cada uno de los clasificadores entrenados, con las instancias formadas por las medidas aleatorias extraídas de las imágenes, son satisfactorios. Han generalizado bien todos los clasificadores, debido a los atributos de las instancias.

6.5.2 Evaluación Modelos de Medidas Estadísticas

En la Tabla 19 se muestran los resultados de la evaluación de los modelos entrenados con medidas estadísticas:

CLASIFICADOR	TIEMPO EJECUCIÓN	% ACIERTOS	INSTANCIAS CORRECTAMENTE CLASIFICADAS	INSTANCIAS INCORRECTAMENTE CLASIFICADAS	ERROR ABSOLUTO	MEDIA FALSOS POSITIVOS
NaiveBayes	2s	73.0833%	877	323	0.2694	0.269
Logistic	2s	100%	1200	0	0%	0
MultilayerPerceptron	2s	100%	1200	0	0.0008	0
RBFNetwork	3s	63.4167%	761	439	0.383	0.366
SMO	2s	100%	1200	0	0	0
VotedPerceptron	2s	100%	1200	0	0	0
IB1	23s	100%	1200	0	0	0
IBk	7s	100%	1200	0	0.0005	0
KStar	8m,23s	79.6667%	956	244	0.1954	0.203
LWL	3m,41s	100%	1200	0	0	0
J48	2s	100%	1200	0	0	0
REPTree	1s	100%	1200	0	0	0

Tabla 19: Resultados de la evaluación de modelos con medidas estadísticas

Se puede observar que hay tres clasificadores que están muy por debajo en eficacia. Sus porcentajes de aciertos no llegan al 80% ni en el mejor caso. Durante la evaluación del entrenamiento ya se observaba que para NaiveBayes, RBFNetwork y KStar se obtenían peores resultados que con el resto. Estos resultados, más negativos, que con las medidas aleatorias, pueden significar que la naturaleza de los datos (medidas estadísticas) provoca que algunos clasificadores no generalicen bien. Por tanto, se puede deducir que las medidas estadísticas son peores, que las aleatorias, para predecir si una imagen tiene información ocultada con Vecna.

6.5.3 Evaluación Modelos de Medidas Estegoanalíticas

En la Tabla 20 se muestran los resultados de la evaluación de los modelos entrenados con medidas estegoanalíticas:

CLASIFICADOR	TIEMPO EJECUCIÓN	% ACIERTOS	INSTANCIAS CORRECTAMENTE CLASIFICADAS	INSTANCIAS INCORRECTAMENTE CLASIFICADAS	ERROR ABSOLUTO	MEDIA FALSOS POSITIVOS
NaiveBayes	1s	82.1667%	986	214	0.1833	0.178
Logistic	1s	96.75%	1161	39	0.0312	0.033
MultilayerPerceptron	2s	100%	1200	0	0.0008	0
RBFNetwork	2s	100%	1200	0	0	0
SMO	1s	100%	1200	0	0	0
VotedPerceptron	1s	100%	1200	0	0	0
IB1	15s	100%	1200	0	0	0
IBk	6s	100%	1200	0	0.0005	0
KStar	13m,47s	85.25%	1023	177	0.1747	0.148
LWL	3m,52s	100%	1200	0	0	0
J48	1s	100%	1200	0	0	0
REPTree	1s	100%	1200	0	0	0

Tabla 20: Resultados de la evaluación de modelos con medidas estegoanalíticas

Los resultados obtenidos por las medidas estegoanalíticas son mejores que los obtenidos por las medidas estadísticas, pero peores que los obtenidos por las medidas aleatorias. Durante la evaluación en el entrenamiento ya se observó que había ciertos clasificadores que generalizaban peor, como NaiveBayes, o KStar. Y de la misma forma sucede en la evaluación frente a instancias nuevas.

6.5.4 Evaluación Modelos de Todas las Medidas

En la Tabla 21 se muestran los resultados de la evaluación de los modelos entrenados con todas las medidas anteriores:

CLASIFICADOR	TIEMPO EJECUCIÓN	% ACIERTOS	INSTANCIAS CORRECTAMENTE CLASIFICADAS	INSTANCIAS INCORRECTAMENTE CLASIFICADAS	ERROR ABSOLUTO	MEDIA FALSOS POSITIVOS
NaiveBayes	3s	100%	1200	0	0	0
Logistic	3s	100%	1200	0	0	0
MultilayerPerceptron	5s	100%	1200	0	0.0003	0
RBFNetwork	4s	100%	1200	0	0	0
SMO	3s	100%	1200	0	0	0
VotedPerceptron	3s	100%	1200	0	0	0
IB1	57s	100%	1200	0	0	0
IBk	30s	100%	1200	0	0.0005	0
KStar	42m,32s	100%	1200	0	0.0004	0
LWL	14m,8s	99.9167%	1199	1	0.0008	0.001
J48	2s	100%	1200	0	0	0
REPTree	2s	100%	1200	0	0	0

Tabla 21: Resultados de la evaluación de modelos con todas las medidas

Estos resultados son satisfactorios, aunque era de esperar. Mediante el uso de todas las medidas se han conseguido unos resultados comparables a los de las medidas aleatorias. Como ya se pudo comprobar en la evaluación del entrenamiento, esto es debido a que los mejores atributos seleccionados son en su mayor parte medidas aleatorias. Con los resultados del entrenamiento se pensó que la mejor solución sería dada por las medidas aleatorias, y finalmente así ha sido. Además, al ser una técnica LSB, se podía sospechar en un principio que los mejores resultados los darán las medidas aleatorias Esto es debido a las modificaciones, que realiza LSB, sobre la información de la imagen.

6.6 Selección Mejor Modelo

A la vista de todos los resultados se debe elegir un modelo entrenado con las instancias de las medidas aleatorias de las imágenes. El mejor modelo se ha implementado en JUBSAC como estegoanalizador para poder analizar imágenes y detectar si tienen información oculta. La elección de este analizador, además permitirá obtener un estegoanalizador más eficiente, pues necesitará menos medidas a calcular al haber elegido las medidas aleatorias frente a todas las medidas. Ya que, para las medidas aleatorias, todos los modelos clasifican con un 100% de aciertos en validación, pese a que Weka muestre algún pequeñísimo error medio de menos de 1 milésima, despreciable, será difícil elegir el mejor modelo. Como el mejor modelo será implementado, se deberá de optimizar al máximo el tiempo de ejecución. Por esta razón se han descartado algunos modelos lentos, como IB1, IBk, KStar y LWL. Por lo que quedan 8 modelos:

- NaiveBayes.
- Logistic.
- MultilayerPerceptron.
- RBFNetwork.
- SMO.
- VotedPerceptron.
- J48
- REPTree.

Todos estos clasificadores han obtenido unos modelos equivalentes para estegoanalizar imágenes, sin diferencias en sus resultados frente a imágenes nuevas. En las publicaciones de estegoanálisis con IA, se observa en repetidas ocasiones el uso de las redes neuronales con el algoritmo de la retropropagación. Esto sólo se da en el perceptrón multicapa (Multilayer Perceptron). Así que finalmente se ha decidido que el modelo que va a usarse como estegoanalizador en JUBSAC es el *MultilayerPerceptron*. Aunque esté no es el único mejor clasificador, se ha decidido que es el mejor modelo para su implementación debido a su reconocimiento como modelo frecuentemente usado en la inteligencia artificial.

7. CONCLUSIONES Y LÍNEAS FUTURAS

7.1 Conclusiones

Durante este proyecto se ha desarrollado la herramienta llamada JUBSAC. Esta implementa la generalización de un marco único para la creación de estegoanalizadores. JUBSAC crea otros programas construidos gracias a la inteligencia artificial. Es un creador de analizadores esteganográficos universales, o a ciegas, para imágenes.

El carácter experimentador de JUBSAC permite que sea usado como una herramienta para crear nuevos métodos estegoanalíticos, o medir la robustez de nuevas, o ya existentes, herramientas esteganográficas. JUBSAC no sólo permite crear estegoanalizadores, sino que se pueden usar y además implementa ya uno, creado por ella misma, para que pueda ser usada la aplicación, simplemente, como un analizador esteganográfico. Este estegoanalizador ha sido fruto de un proceso de experimentación, donde se ha elegido el mejor de una serie de experimentos para implementarle. Dicho estegoanalizador tiene una frecuencia de aciertos del 100%, en las evaluaciones realizadas con él. Con esta tasa de aciertos tan elevada este analizador es capaz de romper el algoritmo, usado por la herramienta esteganográfica Vecna, para la ocultación de información en imágenes. Dicho algoritmo no estaba roto previamente, y gracias al uso de JUBSAC, se ha conseguido.

Durante la implementación se ha puesto mucho énfasis para construir una interfaz que sea fácilmente entendible y usable, con un diseño cómodo y atractivo a la vista para asegurar una mejor experiencia a los usuarios. Intentando dar transparencia a todos los procesos, que se dan por debajo de la interfaz, para homogeneizar la herramienta al máximo.

La herramienta es, en su mayor medida, para la investigación de nuevas técnicas estegoanalíticas. Esto hace que se espere de ella una vida útil más allá de la entrega de este proyecto académico. Y no sólo que sea usada, sino que sea ampliada y mejorada, para que ofrezca a los usuarios un amplio abanico de herramientas para la creación y prueba de estegoanalizadores, no sólo de imágenes, sino de otro tipo de archivos.

Las opciones que da la herramienta, son muy amplias, y realizar experimentos con todas las combinaciones, no ha sido posible, debido a que se necesita tiempo y recursos. Aún así si se han probado todas las opciones como para asegurar que funcionan adecuadamente.

7.2 Líneas Futuras

Si bien la aplicación ofrece una serie de posibilidades muy amplias, durante el desarrollo se han detectado posibles funcionalidades que no han sido implementadas por falta de tiempo:

1. Traducción al inglés de la interfaz, del código, del manual de usuario, del manual de cómo ampliar la herramienta, para la internalización total de la herramienta.

2. Elaboración de unos experimentos de mayor tamaño (≈ 40000 imágenes de partida) con unas máquinas más potentes, probando todas las opciones de la herramienta y algoritmos, para finalmente evaluar todas las soluciones y utilizar la mejor solución alcanzada.

3. Publicación de la herramienta, junto con la licencia que se estime oportuna, en sitios web de software gratuito y/o académico.

4. Realizar ampliaciones surgidas durante la fase previa de experimentación, o propuestas por diversos usuarios. Como añadir nuevas medidas en la extracción de medidas, y/o nuevas herramientas esteganográficas, y/o nuevos clasificadores, y/o nuevos formatos como música, videos, etc.

5. Creación de una página web a la herramienta, para publicar todas sus versiones, manuales, artículos, etc.

6. Ampliación de la herramienta para que soporte más sistemas operativos. Para así poder evaluar su rendimiento al ejecutarlo en máquinas con otros entornos que no sean Windows. Y acceder a herramientas que sólo están para otros sistemas.

7. Paralelización de más tareas, no sólo la ejecución de herramientas esteganográficas y la ejecución de experimentos con Weka.

8. Intentar detectar la longitud del mensaje, o la técnica usada.

8. GESTIÓN DEL PROYECTO

En este capítulo se explican todas las tareas para el logro del proyecto. Especifica las fases del desarrollo del proyecto, los medios empleados, la planificación del proyecto y el presupuesto de la realización del proyecto.

8.1 Fases del Desarrollo

El desarrollo del proyecto se ha realizado en las siguientes fases y subfases:

- **Documentación sobre el estado del arte**:
  - **Técnicas Esteganográficas**: En esta subfase se han estudiado las herramientas de ocultación, profundizando en la ocultación de información en imágenes.
  - **Estegoanálisis e Inteligencia Artificial**: En esta subfase se ha estudiado el estegoanálisis en imágenes, profundizando más en las que hacían uso de la inteligencia artificial para construir estegoanalizadores.
- **Creación de un marco único:** Esta fase se centra en la elaboración de un marco para la creación de estegoanalizadores de imágenes, usando la documentación estudiada previamente, e intentando generalizar todas las técnicas de creación de estegoanalizadores para imágenes.
- **Desarrollo de la aplicación:** Durante esta fase se ha implementado el marco único creado previamente para desarrollar experimentos.
- **Experimentación:** Esta fase corresponde a la elaboración de una serie de experimentos para probar la herramienta y desarrollar un estegoanalizador.
- **Implementación de un Estegoanalizador:** En esta fase se implementa el mejor estegoanalizador surgido durante la fase de experimentación.
- **Redacción de la memoria:**
  - **Documento**: Se realiza, en esta fase, la escritura del presente documento.
  - **Manual del usuario**: Durante esta fase se procede a la escritura de un manual para usar la aplicación.
  - **Manual del programador**: En esta fase se realiza la escritura de un manual para ampliar la aplicación.
- **Realización de una presentación:**
  - **Documento:** Se procede, en esta fase, al desarrollo de una presentación para la defensa del proyecto.
  - **Videos:** En esta última fase se procede a la grabación de una serie de videos demostrativos de la aplicación siendo usada.

A continuación, se muestra en la Figura 41, el correspondiente Diagrama de Gantt:

Nombre de tarea	Duración	Comienzo	Fin
Documentación	7 días	mar 15/06/10	mar 22/06/10
Creación Marco Único	5 días?	mié 23/06/10	mar 29/06/10
Desarrollo de la aplicación	57 días	mié 30/06/10	mié 15/09/10
Experimentación	12 días?	jue 16/09/10	jue 30/09/10
Implementación Estegoanalizador	2 días?	vie 01/10/10	sáb 02/10/10
MEMORIA	**17 días?**	**dom 03/10/10**	**mié 20/10/10**
Documento	12 días?	dom 03/10/10	vie 15/10/10
Manual de Usuario	2 días?	sáb 16/10/10	dom 17/10/10
Manual del Programador	3 días?	lun 18/10/10	mié 20/10/10
PRESENTACIÓN	**5 días?**	**jue 21/10/10**	**mar 26/10/10**
Documento Presentación	2 días?	jue 21/10/10	sáb 23/10/10
Videos	3 días?	dom 24/10/10	mar 26/10/10

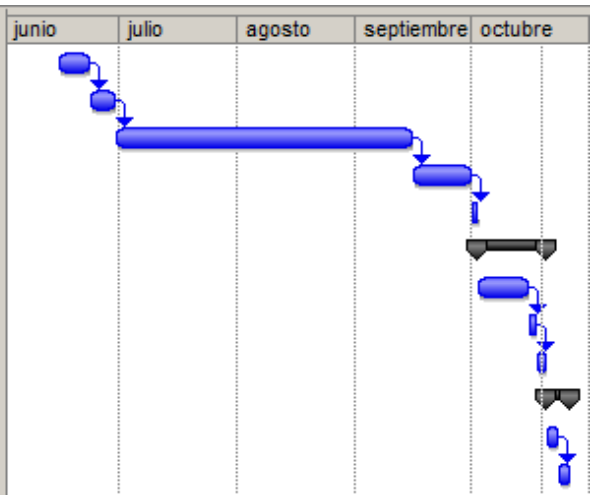

Figura 41: Planificación y diagrama de Gantt

8.2 Medios Empleados

El medio principal más empleado ha sido el relacionado con los recursos temporales. Se ha necesitado mucho tiempo, para elaborar proyecto. Aunque su elaboración haya durado sólo cinco meses, la media de trabajo es de 9 horas por día.

En cuanto a los medios físicos, para hardware sólo ha sido necesario un ordenador portátil para desarrollar la herramienta, y un ordenador de sobremesa, con varios núcleos de procesamiento, para realizar pruebas experimentales de la herramienta.

Ordenador portátil para la programación de JUBSAC:

- Procesador: Intel Celeron 1,86GHz.
- Memoria RAM: 2,00 GB.
- Sistema Operativo: Windows Vista Home Basic de 32 bits.

Ordenador de sobremesa para la experimentación de JUBSAC:

- Procesador: AMD Athlon 64 x2 Dual Core Processor 2,10 GHz.
- Memoria RAM: 1,00 GB.
- Sistema Operativo: Windows Vista Home Basic de 32 bits.

No se ha empleado ninguna máquina especialmente potente, pero en un futuro próximo será usada, ya que así se podrán ejecutar experimentos de grandes dimensiones en menos tiempo. Otros medios físicos son una conexión ADSL (Telefónica 6 MB) a Internet y en ocasiones usar la red VPN de la Universidad Carlos III, para poder tener acceso libre a artículos y publicaciones necesarias para la fase de documentación del proyecto.

Los medios software han sido todos gratuitos (AdobeReader, FireFox, NetBeans, Notepad++ y Weka) o de pago (Windows Vista y Micosoft Office con licencia universitaria) o versiones de prueba (AdobePDFMaker, Camtasia Studio y NeoDownloader), sólo necesitando tiempo y esfuerzo para conseguir los necesarios y usarlos adecuadamente. También se han usado servidores de correo gratuitos (GMail) para comunicarse con autores, de ciertas herramientas esteganográficas, para pedir su ayuda con estas. Incluso servidores gratuitos de almacenamiento de archivos (DropBox) para guardar y compartir el trabajo desarrollado.

8.3 Presupuesto

A continuación, se detalla el presupuesto[15]:

[15] Plantilla de presupuesto: Fichero en excel con un modelo para ayudar a confeccionar el presupuesto del proyecto fin de carrera.
<http://www.uc3m.es/portal/page/portal/administracion_campus_leganes_est_cg/proyecto_fin_carrera/Formulario_PresupuestoPFC-TFG%20(3)_2.xlsx> [Octubre de 2010]

UNIVERSIDAD CARLOS III DE MADRID
Escuela Politécnica Superior

PRESUPUESTO DE PROYECTO

1.- Autor:

Javier García-Cuerva Velasco

2.- Departamento:

Escuela Politécnica Superior, Ingeniería Informática, Departamento de Informática

3.- Descripción del Proyecto:

- Titulo: **Creación de una herramienta para la generación de analizadores esteganográficos para imágenes**
- Duración (meses): **5**
- Tasa de costes Indirectos: **20%**

4.- Presupuesto total del Proyecto (valores en Euros):

39.664,00 Euros

5.- Desglose presupuestario (costes directos)

PERSONAL

Apellidos y nombre	N.I.F. (no rellenar - solo a titulo informativo)	Categoría	Dedicación (hombres mes) [a]	Coste hombre mes	Coste (Euro)	Firma de conformidad
Blasco Alís, Jorge		Consultor	1	4.289,54	4.289,54	
García-Cuerva Velasco, Javier		Ingeniero	10,3	2.694,39	27.752,22	
					0,00	
					0,00	
					0,00	
		Hombres mes	**11,3**	**Total**	32.041,76	

9 horas de media al día durante 150 días (5 meses): 9*150=1350 horas;1350/131,25=10,3 hombres me

[a] 1 Hombre mes = 131,25 horas. Máximo anual de dedicación de 12 hombres mes (1575 horas

Máximo anual para PDI de la Universidad Carlos III de Madrid de 8,8 hombres mes (1.155 horas)

EQUIPOS

Descripción	Coste (Euro)	% Uso dedicado proyecto	Dedicación (meses)	Periodo de depreciación	Coste imputable [d]
Intel Celeron 1,86GHz	500,00	100	5	60	41,67
AMD Athlon Dual Core 2.10GHz	600,00	100	2	60	20,00
		100		60	0,00
		100		60	0,00
		100		60	0,00
					0,00
				Total	61,67

[d] Fórmula de cálculo de la Amortización:

$$\frac{A}{B} \text{xCxD}$$

A = nº de meses desde la fecha de facturación en que el equipo es utilizado
B = periodo de depreciación (60 meses)
C = coste del equipo (sin IVA)
D = % del uso que se dedica al proyecto (habitualmente 100%)

SUBCONTRATACIÓN DE TAREAS

Descripción	Empresa	Coste imputable
	Total	0,00

OTROS COSTES DIRECTOS DEL PROYECTO[e]

Descripción	Empresa	Costes imputable
Dietas	Restaurantes	400,00
Office Académico 07	Microsoft	52,00
Transporte	BP	100,00
ADSL	Telefónica	200,00
Material Oficina	Papelerias	50,00
Luz	Iberdrola	200,00
	Total	1.002,00

[e] Este capítulo de gastos incluye todos los gastos no contemplados en los conceptos anteriores, por ejemplo: fungible, viajes y dietas, otros,...

6.- Resumen de costes

Presupuesto Costes Totales	**Presupuesto Costes Totales**
Personal	32.042
Amortización	62
Subcontratación de tareas	0
Costes de funcionamiento	1.002
Costes Indirectos	6.621
Total	39.727

El presupuesto total de este proyecto asciende a la cantidad de 39727 euros.

Leganés a 10 de Octubre de 2010

El ingeniero proyectista

Fdo. Javier García-Cuerva Velasco

9. GLOSARIO

A continuación, se dará la correspondencia de las siglas y acrónimos mencionados durante la redacción de la memoria:

- **AES:** Advanced Encryption Standard.
- **ANSI:** American National Standards Institute.
- **ANOVA:** Analysis of Variance
- **BMP:** BitMaP.
- **BSD**: Berkeley Software Distribution
- **CBC:** Cipher Block Chaining.
- **CFB:** Cipher Feedback.
- **DCT:** Discrete Cosine Transform.
- **DFT:** Discrete Fourier Transform
- **DOS:** Disk Operating System
- **DWT:** Discrete Wavelet Transform.
- **EXE:** Executable File.
- **GIF:** Graphics Image Format.
- **GPL:** GNU General Public License.
- **GUI:** Graphical User Interface.
- **HTML:** HyperText Markup Language.
- **IA:** Inteligencia Artificial.
- **IQM:** Image Quality Metrics
- **JPEG:** Joint Photographic Experts Group.
- **LSB:** Least-Significant Bit.
- **MS-DOS:** MicroSoft Disk Operating System
- **PGP:** Pretty Good Privacy.
- **PNG:** Portable Network Graphics.
- **RGB:** Red Green Blue.
- **SDS:** Spatial Domain Steganography.
- **TCP/IP:** Transmission Control Protocol/Internet Protocol.
- **TDS**: Transform Domain Steganography.
- ***XML*:** Extensible Markup Language.

10. REFERENCIAS

[1] Miguel Saiz Serrano, *"Conferencias FIST",* Madrid, Septiembre de 2007.

[2] G. J. Simmons, "The prisoner's problem and the subliminal channel", *Advances in Cryptology*, CRYPTO '83, D. Chaum, ed., Plenum Press, 1984, 51-67.

[3] Arturo Ribagorda G., Juan Estévez-Tapiador y Julio Hernández, *"Esteganografía, Esteganálisis e Internet. Descubriendo el reverso de Internet: web mining, mensajes ocultos y secretos aparentes*", Madrid, 22 de Febrero de 2007.

[4] Roberto Gómez Cárdenas, "*Tercer Congreso de Seguridad en Cómputo*", Madrid, 29 de Octubre de 2008.

[5] Neil F. Johnson, Sushil Jajodia, "Exploring Steganography: Seeing the Unseen", *Computing Practices '98*, pp: 26-34, George Mason University, 1998. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.40.460&rep=rep1&type=pdf> [Junio de 2010]

[6] V. Kavitha y K.S. Easwarakumar, "Neural Based Steganography", PRICAI 2004, LNAI 3157, pp. 429–435, 2004.

[7] Fernando C. Gonzalez, "Counter Terrorist Steganography Search Engine", Cranfield University, 2002.

[8] Tomas Pevny, JessicaFridrich, "Novelty Detection in Blind Steganalysis", *International Multimedia Conference, Proceedings of the 10th ACM workshop on Multimedia and security*, pp: 167-176, Oxford, United Kingdom, 2008. <http://portal.acm.org/citation.cfm?id=1411357>

[9] Roshidi Din and Azman Samsudin. "Digital Steganalysis: Computacional Intelligence Approach*", International Journal of Computers*, *Issue 1, Volume 3*, 30/1/2009. <www.naun.org/journals/computers/ijcomputers-130.pdf> [Junio de 2010]

[10] USA Today, "Terrorist instructions hidden online" y "*Terror groups hide behind Web encryption*", 5/2/2001.

[11] New York Times*,* "al-Qaeda had used steganography to encode messages into images to *prepare and execute the September 11, 2001 Terrorist Attack",* Octubre de 2001.

[12] Duncan Campbell, "How the Terror Trail Went Unseen", Telepolis, 10/08/2001. <http://www.heise.de/tp/english/inhalt/te/9751/1.html> [Junio de 2010]

[13] The Federal Plan for Cyber Security and Information Assurance Research and Development, Abril de 2006. <http://www.nitrd.gov/pubs/csia/csia_federal_plan.pdf> [Junio de 2010]

10. REFERENCIAS

[14] Stephen Brown, *Front Page Magazine.* <http://jeffersonsrebels.blogspot.com/2010/01/sexual-depravity-breeds-savage.html> [Junio de 2010]

[15] The Federal Bureau of Investigation, 2010. <http://www.justice.gov/opa/documents/062810complaint2.pdf> [Junio de 2010]

[16] Sally Adee , "Spy vs Spy", *IEEESpectrum*, Agosto de 2008. <http://spectrum.ieee.org/computing/software/spy-vs-spy> [Junio de 2010]

[17] Niels Provos Peter Honeyman, Detecting Steganographic Content on the Internet, *CITI Technical Report 01-11,* 31/8/2001. <http://www.citi.umich.edu/u/provos/papers/detecting.pdf> [Junio de 2010]

[18] Kriptópolis, "Contra esteganografía... doble esteganografía",23/8/2008. <http://www.kriptopolis.org/doble-esteganografia?page=2> [Junio de 2010]

[19] John McCarthy, "What is Artificial Intelligence?", 12 de Noviembre de 2007.

[20] Vilalta R. and Drissi Y. (2002). "A perspective view and survey of meta-learning", *Artificial Intelligence Review*, 18(2), 77—95.

[21] COMP9414 Artificial Intelligence, "The Machine Learning Dictionary", University of New South Wales, Sydney, 12 de Octubre de 2010.

[22] Ingemar J. Cox, Matthew L. Miller, Jeffrey A. Bloom, Jessica Fridrich and Ton Kalker, "Digital Watermarking and Steganography" (Second Edition), Morgan Kaufmann, 2008.

[23] Google Académico, "Steganography", Octubre de 2010. <http://scholar.google.es/scholar?hl=es&q=steganography&lr=&as_ylo=2007&as_vis=0> [Octubre de 2010]

[24] Samir Vaidya, "OpenStego v0.5.2",31/3/2009. <http://openstego.sourceforge.net/> [Julio de 2010]

[25] Free Software Foundation, "GNU General Public License 2.0 (GPL)", 2007. <http://www.gnu.org/copyleft/gpl.html> [Septiembre de 2010]

[26] Heinz Repp, "Hide4PGP v2.0", 8/12/2003. <http://www.heinz-repp.onlinehome.de/Hide4PGP.htm> [Julio de 2010]

[27] Phil Zimmermann, PGP 8.0 (Pretty Good Privacy), 2002/12/03. <http://www.pgpi.org/> [Septiembre de 2010]

[28] Henry Hastur, "Stealth 1.1". <http://www.cypherspace.org/openpgp/stealth/> <http://www.unicorn.com/pgp/s-readme.html> [Julio de 2010]

[29] Matthew Kwan, GifShuffle, 6/1/2003. <http://www.darkside.com.au/gifshuffle/description.html> [Julio de 2010]

[30] David A. Huffman, "A Method for the Construction of Minimum-Redundancy Codes", *I.R.E.,* Septiembre de 1952, pp. 1098-1102.

[31] Matthew Kwan," Information Concealment Engine (ICE)", 7/6/1999. <http://www.darkside.com.au/ice/index.html> [Septiembre de 2010]

[32] Andreas Westfeld, "F5 v0.1", Dresden, Germany. <http://www.inf.tu-dresden.de/~aw4> [Julio de 2010]

[33] Andreas Westfeld, "High Capacity Despite Better Steganalysis (F5–A Steganographic Algorithm)", *Moskowitz, I.S. (eds.): Information Hiding. 4th International Workshop. Lecture Notes in Computer Science*, Vol.2137. Springer-Verlag, Berlin Heidelberg New York (2001) 289-302. <http://os.inf.tu-dresden.de/~westfeld/publikationen/f5.pdf> [Julio de 2010]

[34] Jessica Fridrich, Miroslav Goljan, Dorin Hogea, "Steganalysis of JPEG Images: Breaking the F5 Algorithm". <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.141.1845&rep=rep1&type=pdf> [Julio de 2010]

[35] Stefan Hetzl, "Steghide v0.5.1". <http://steghide.sourceforge.net/> [Julio de 2010]

[36] Joan Daemen y Vincent Rijmen, "Advanced Encryption Standard (AES)", 26/11/2001.

[37] John Collomosse, "BlindSide BMP Cryptographic Tool, v0.9", 2000. <http://personal.ee.surrey.ac.uk/Personal/J.Collomosse/blindside.php> [Agosto de 2010]

[38] Universidad de Kloblenz – Landau, German, "Vecna - A tool for turning images into secret keepers". <http://www.uni-koblenz.de/~strauss/vecna/> [Septiembre de 2010]

[39] Allan Latham, "JPHIDE y JPSEEK, v0.5", Agosto de 1999. <http://linux01.gwdg.de/~alatham/stego.html> [Julio de 2010]

[40] Derek Upham , "JSteg", Junio de 2007. <http://csis.bits-pilani.ac.in/faculty/murali/netsec-09/seminar/refs/anuroopsrep.pdf> [Julio de 2010]

[41] Niels Provos , "OutGuess v0.2", Febrero de 2001. <http://www.outguess.org/> [Julio de 2010]

[42] Universidad de California, "BSD (Berkeley Software Distribution)", 1989. <http://www.opensource.org/licenses/bsd-license.php> [Septiembre de 2010]

[43] Fabian Hansmann, "STEGANOS v1.4", Frankfurt, Alemania, 10/7/1996.

[44] Eduardo Sztokbant, "StegTools, v0.4c", 6/6/2005. <http://sourceforge.net/projects/stegotools/> [Julio de 2010]

[45] Kathryn Hempstalk, "Digital Invisible Ink Toolkit". <http://diit.sourceforge.net/doco.html > <http://sourceforge.net/projects/diit/> [Julio de 2010]

[46] Nathanaël Cottin, "Hide & Reveal v1.7.0", 28/5/2010. <http://hidereveal.ncottin.net/> [Julio de 2010]

[47] Colin Moroney , “Hide and Seek, v4.1”.
<http://packetstormsecurity.org/crypt/stego/DOS/> [Julio de 2010]

[48] Xuejia Lai y James L. Massey, “IDEA, International Data Encryption Algorithm”, Escuela Politécnica Federal de Zúrich, 1991. *[Septiembre de 2010]*

[49] Ísmail Avcibas, Bülent Sankur, Khalid Sayood, “Statistical evaluación of image quality measures“, *Journal of Electronic Imaging,* vol.11(2), Abril de 2002.

[50] Ísmail Avcibas, Nasir Memon, Bülent Sankur, “Steganalysis Using Image Quality Metrics”, *IEEE Transactions on image processing,* vol. 12, nº 2, Febrero de 2003.

[51] Ísmail Avcibas, Nasir Memon, Bülent Sankur, “Steganalysis Base don Image Quality Metrics”, *2001*.

[52] Ahmet M. Eskicioglu, Paul S. Fisher, “Image Quality Measures and Their Performance”, *IEEE Transactions on Comunications*, vol. 43, nº 12, Diciembre de 1995.

[53] S. Liu, H. Yao, y W. Gao, “Neural network based steganalysis in still images”, *Proceedings of the 2003 International Conference on Multimedia and Expo (ICME 2003*), vol.2, pp. II – 509-512, Julio de 2003.

[54] Yuan Liu, Li Huang, Ping Wang, Guodong Wang, “A Blind Image Steganalysis Based on Features from Three Domains”, *Chinese Control and Decision Conference (CCDC 2008*), pp. 2933-2936, 2008.

[55] S. Geetha, Siva S. Sivatha Sindhu, N. Kamaraj, “Detection of Stego Anomalies in Images Exploiting the Content Independent Statical Footprints of the Steganograms”, Informatica 33 (2009) 25-40,1/9/2008.

[56] S. Geetha, Siva S. Sivatha Sindhu, N. Kamaraj, “Evolving GA Classifier for breaking the Steganographic Utilities: Stools, Steganos and Jsteg””, International Conference on Computational Intelligence and Multimedia Applicatins 2007, Agosto de 2007.

[57] Niels Provos, “Steganography Detection with Stegdetect”.
<http://www.outguess.org/detection.php> *[Julio de 2010]*

[58] Jessica Fridrich, Miroslav Goljan, Dorin Hogea, “Steganalysis of JPEG Images: Breaking the F5 Algorithm”, 5th International Workshop on Information Hiding, pp. 310—323, 2001.

[59] Graeme Bell and Yeuan-Kuen Lee, “A Method for Automatic Identification of Signatures of Steganography Software”, *IEEE Transactions on Information Forensics and Security, vol*. 5, nº 2, Junio de 2010.

[60] Alex Zaharis, Di-Mar, “Ben-4D Steganalysis Software”.
<http://ben4dstegdetect.sourceforge.net/ >
<http://sourceforge.net/projects/ben4dstegdetect/> [Agosto de 2010]

[61] Alfonso Muñoz, “StegSecret. A simple steganalysis tool”.
<http://stegsecret.sourceforge.net/> [Julio de 2010]

10. REFERENCIAS

[62] SARC, “Steganography Analysis and Research Center”. <http://www.sarc-wv.com/> [Agosto de 2010]

[63] WetStone, “Stego Suite”. <http://www.wetstonetech.com/faq_stego.html> [Agosto de 2010]

[64] The University of Waikato (Nueva Zelanda), “Weka 3: Data Mining Software in Java”, 2008. <http://www.cs.waikato.ac.nz/ml/weka/> [Agosto de 2010]

[65] Jason Hunter, “JDOM 1.1.1”, 26/7/09. <http://www.jdom.org/> [Agosto de 2010]

[66] Stephen Manley, ”A Java implementation of the Discreet Cosine Transform”. <http://eagle.uccb.ns.ca/steve/home.html> [Agosto de 2010]

[67] Christoph Lauer, “Código Original en C de Mike Jackson”. <University of Edinburgh 1999-2001> [Agosto de 2010]

[68] Hyperic, Inc , “Hyperic SIGAR API”, 2010. <http://support.hyperic.com/display/SIGAR/Home> [Agosto de 2010]

[69] John Walker, “ENT, A Pseudorandom Number Sequence Test Program“, 28/1/2008.

[70] J. J. Fridrich, M. Goljan. “Practical steganalysis of digital images - state of the art”. *Security and Watermarking of Multimedia Contents, SPIE* vol.4675, pages 1-13, 2002.

[71] J. Fridrich, M. Goljan, y R. Du, "Reliable Detection of LSB Steganography in Grayscale and Color Images ", *Proc. ACM, Special Session on Multimedia Security and Watermarking*, Ottawa, Canada, pp. 27–30, 5/10/2001.

[72] J. Fridrich, M. Goljan, y R. Du, “Detecting LSB Steganography in Color and Gray-Scale Images”, Magazine of IEEE Multimedia, Special Issue on Security, pp. 22–28, número de Octubre-Diciembre de 2001.

[73] Sorina Dumitrescu, Xiaolin Wu, y Zhe Wang, "Detection of LSB steganography via Sample Pair analysis", *IEEE Transaction on Signal Processing*, Vol. 51, Nº 7, Julio del 2003.

[74] J. Fridrich, R. Du, M. Long. “Steganalysis of lsb encodign in color images”. *Proccedings of ICME 2000,* New York City, 31 Julio a 2 de Agosto.

[75] Y. Freund, R. E. Schapire, “Large margin classification using the perceptron algorithm”, *11th Annual Conference on Computational Learning Theory*, New York, NY, 209-217, 1998.

[76] Neowise Software, “NeoDownloader, formerly Express WebPictures”, 2010. <http://www.neowise.com/neodownloader/> [Junio de 2010]

[77] Free Stock Photography, <http://www.adigitaldreamer.com/gallery/> [Junio de 2010]

[78] Cepolina Photo, <http://www.cepolina.com/freephoto/> [Junio de 2010]

[79] Turbo Photo, <http://www.turbophoto.com/Free-Stock-Images/> [Junio de 2010]

[80] Rodrigo Reyes, “JSmooth, a Java Executable Wrapper”, 20/5/2007. <http://jsmooth.sourceforge.net/> [Septiembre de 2010]

Anexo 1. MANUAL DE USUARIO

En este primer anexo se especifica el manual de uso de JUBSAC.

1. Requisitos Mínimos

Para poder ejecutar JUBSAC (Java Universal Blind StegAnalyzer Creator) se necesitan obligatoriamente los siguientes requisitos:

- Windows XP SP3 o superior.
- Java 1.5 o superior.

Si no se cumplen estos requisitos la aplicación no podrá empezar a funcionar. Si se ejecuta en un entorno distinto de Windows, avisará de ello y se cerrará. Si no tiene instalado Java, se avisará para poder instalarlo. Si la versión de Java es anterior a la 1.5, la herramienta no funcionará. Una vez que los requisitos obligatorios están cumplidos, para poder ejecutar JUBSAC con normalidad se recomiendan los siguientes requisitos mínimos:

- 1 GB de Memoria RAM.
- Resolución de Pantalla de al menos 1280x800.
- Procesador de 1,86 Ghz.
- 100 MB libres de Memoria en el Disco Duro.

Si no se cumplen estos requisitos mínimos no se garantiza que la aplicación funcione adecuadamente. La aplicación como máximo usará 256 MB por sí sola para la JVM (Java Virtual Machine) y si no tuviera suficiente Memoria RAM para esos 256 MB, y los necesitara, la JVM dará una excepción y se parará. La aplicación es un archivo ejecutable de Windows (exe). Para ejecutarlo, sólo hay que hacer doble clic sobre él. Este ejecutable ha sido creado gracias a un programa que crea envoltorios exes de archivos jar (JSmooth [80]). Si lo desea también puede ejecutar el jar con JAVA. Si va a utilizar un gran número de imágenes se recomienda ejecutar usando el siguiente comando:

```
java –Xmx256M –jar JUBSAC.jar
```

En diversos módulos de la aplicación el usuario puede elegir cuanta Memoria RAM libre dar, como máximo, al proceso que va a ejecutar en ese módulo. La aplicación muestra al usuario de cuanta memoria libre dispone en ese momento. Pero la aplicación no dejará al usuario elegir más memoria RAM de la que tiene libre. Si el usuario eligiese toda la memoria RAM libre y el proceso que la usa, la necesitase, es posible que el ordenador se paralizara o se bloquee. Ya que en ese momento el sistema operativo va a usar parte de la memoria del disco duro como si fuera memoria principal para ejecutar el proceso, es decir va a usar su memoria virtual. Y puede ocurrir que el tamaño del disco duro para memoria virtual no sea suficiente. Aunque llegar a esta situación es bastante complicado, pero si así fuese no debería usar toda la memoria RAM libre, o si usase toda, deberá aumentar el tamaño del espacio del disco duro que se usa para paginación.

2. Errores

JUBSAC controla la mayoría de errores cometidos, intencionadamente o no, por el usuario o situaciones de error que se pueden dar. Hay tres tipos de situaciones a controlar, que se muestran con tres ejemplos:

- **Nota Preventiva:** Por ejemplo cuando se va a cerrar una ventana donde hay información susceptible de ser perdida por el cierre, se tendrá un aviso como el de la Figura 42.

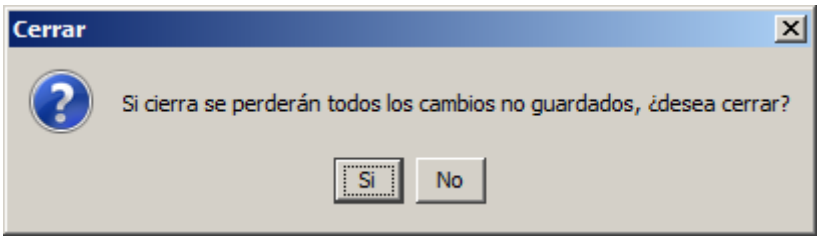

Figura 42: Ventana de nota preventiva

- **Aviso Importante:** Cuando se ejecuta la aplicación se ofrece la recomendación, de la Figura 43, si la resolución no es adecuada.

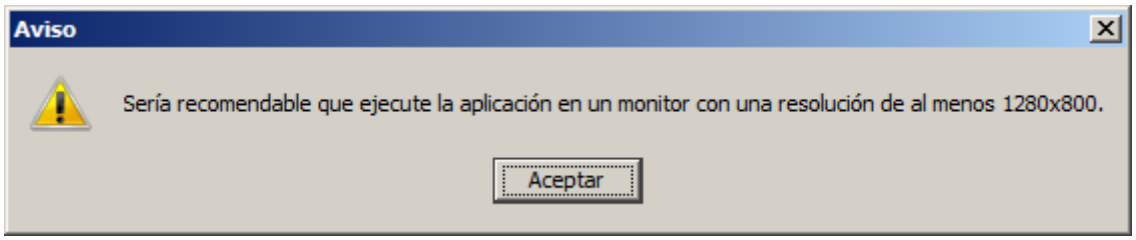

Figura 43: ventana de aviso importante

- **Error Urgente:** Cuando se ejecuta la aplicación y esta no se ejecuta en un sistema Windows, se obtiene un error como el de la Figura 44.

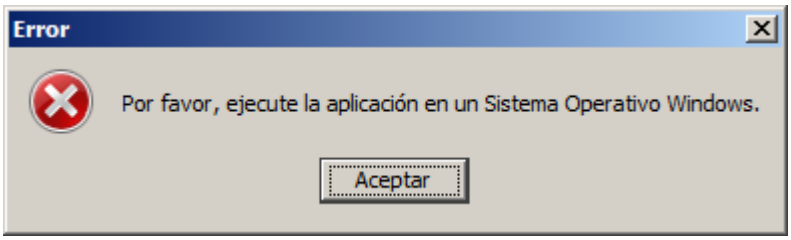

Figura 44: Ventana de error urgente

A continuación, se va a describir cómo usar la aplicación de la forma más detallada y explicativa posible.

3. Menú Principal

El menú principal tiene el aspecto mostrado en la Figura 45:

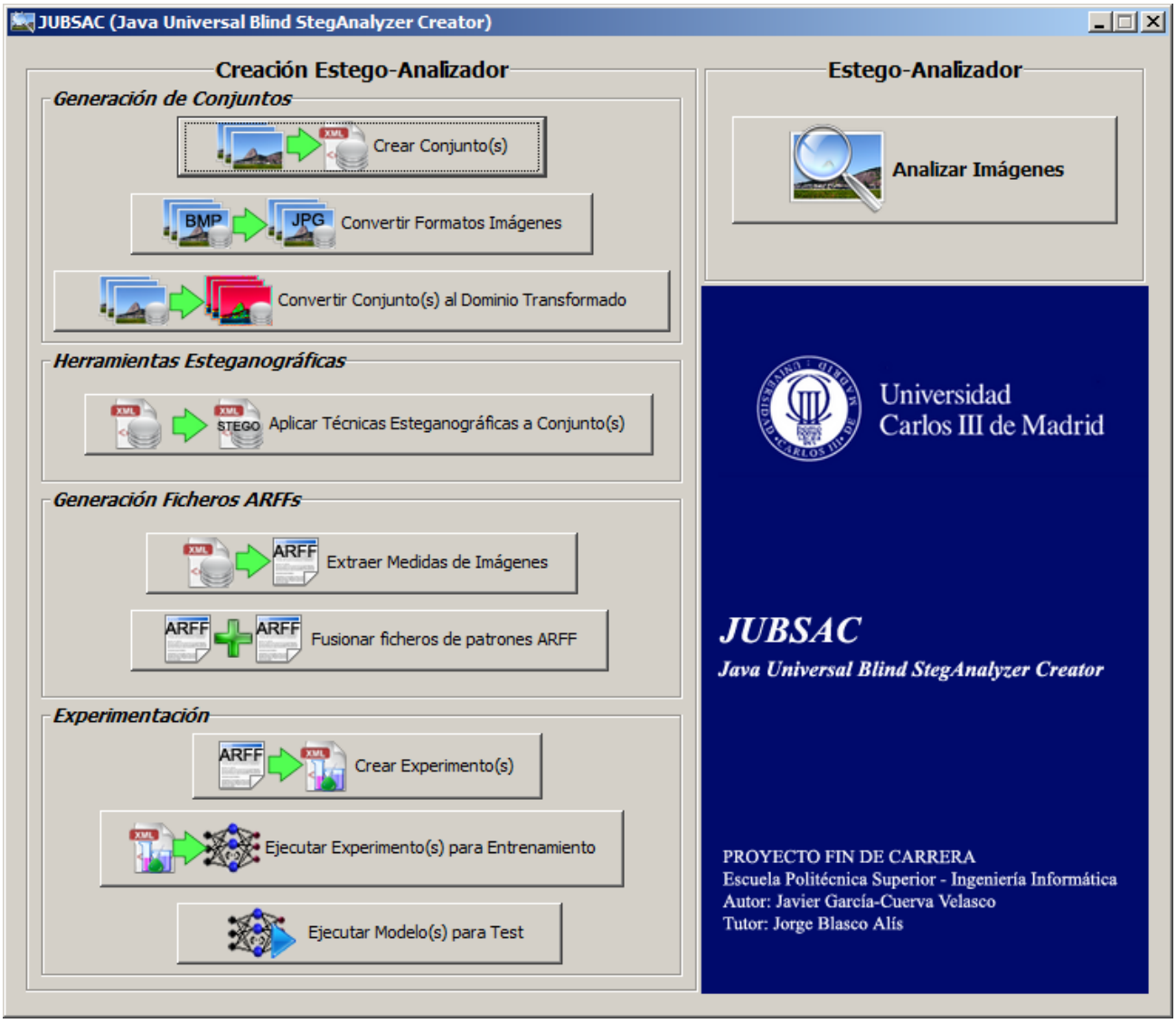

Figura 45: Menú principal

Tiene varias partes claramente diferenciadas:

- **Creación Estego-Analizador**: Incluye las diferentes operaciones que sirven para la creación de un estegoanalizador.
  - Generación de Conjuntos
    - *Crear Conjuntos*
    - *Convertir Formatos Imágenes*
    - *Convertir Conjuntos al Dominio Transformado*

- Herramientas Esteganográficas
  - *Aplicar Técnicas Esteganográficas a Conjuntos*
- Generación Ficheros ARFFs
  - *Extraer Medidas de Imágenes*
  - *Fusionar ficheros de patrones ARFF*
- Experimentación
  - *Crear Experimentos*
  - *Ejecutar Experimentos para Entrenamiento*
  - *Ejecutar Modelos para Test*

- **Estego-Analizador**: Analiza imágenes para buscar información oculta en ella. No hace falta haber creado ninguno previamente.

A continuación se explica cada una de las acciones que se han presentado en el menú. Recordar que si altera una ejecución se podrán dar situaciones de error, para las cuales la aplicación está preparada en la mayoría de los casos.

4. Creación de Conjuntos

En esta ventana se crean conjuntos de imágenes, es decir, se tendrán imágenes ubicadas en directorios (pudiendo ser distintos), siendo referidos desde un solo fichero *XML*. Si se pincha en el botón *Crear Conjunto(s)* se cargará la siguiente ventana, de la Figura 46:

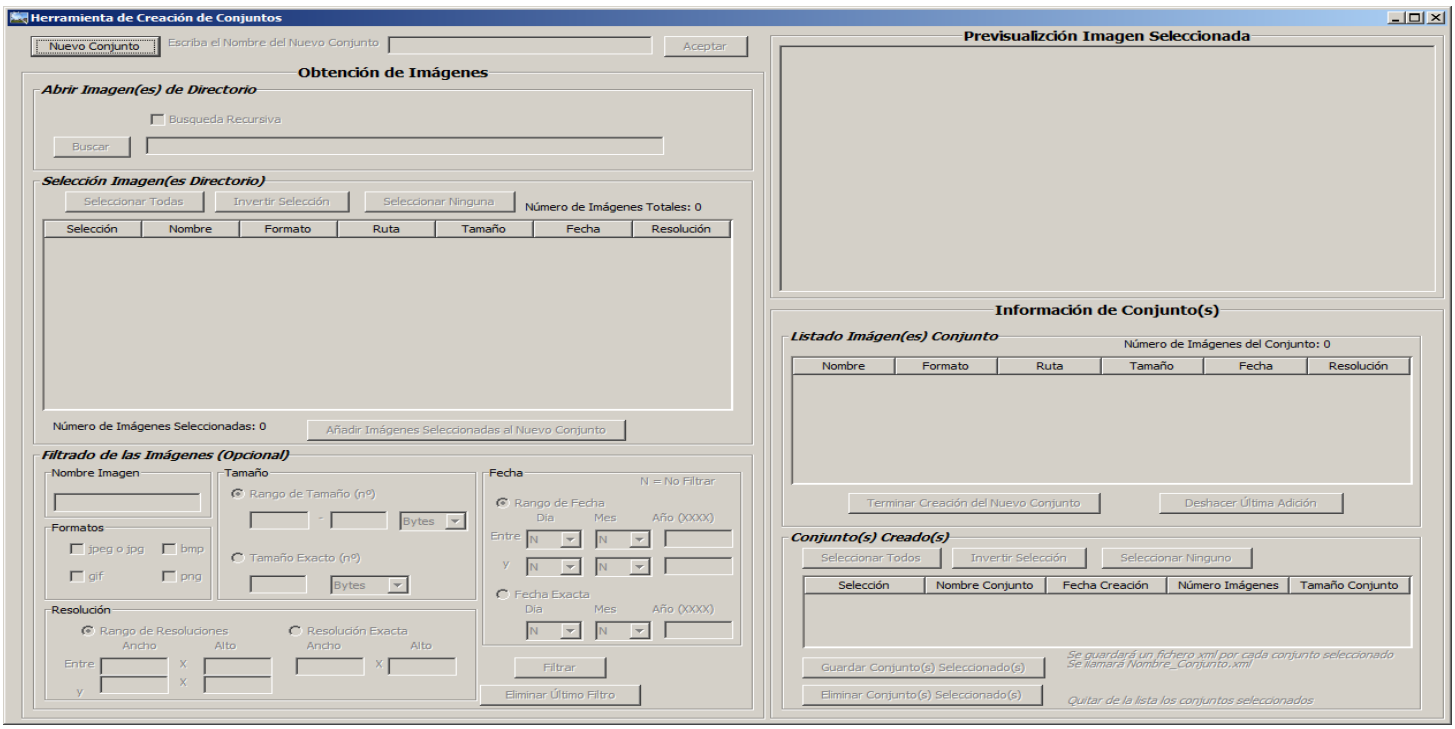

Figura 46: Ventana de la creación de conjuntos

El primer paso para la creación de un conjuntos es clickar al único botón activo, *Nuevo Conjunto,* se deshabilitará este y se situará el foco en el cuadro de texto donde se deberá escribir el nombre del conjunto, en este caso Ejemplo. Se puede ver en la Figura 47:

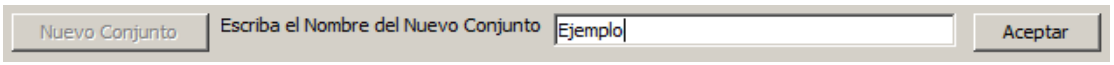

Figura 47: Elección del nombre del nuevo conjunto

4.1 Obtención de Imágenes

Después hacer clic en *Aceptar* o pulsar *Enter o Intro*, en el teclado. Se habilitará una nueva zona de la pantalla. Se puede ver en la Figura 48:

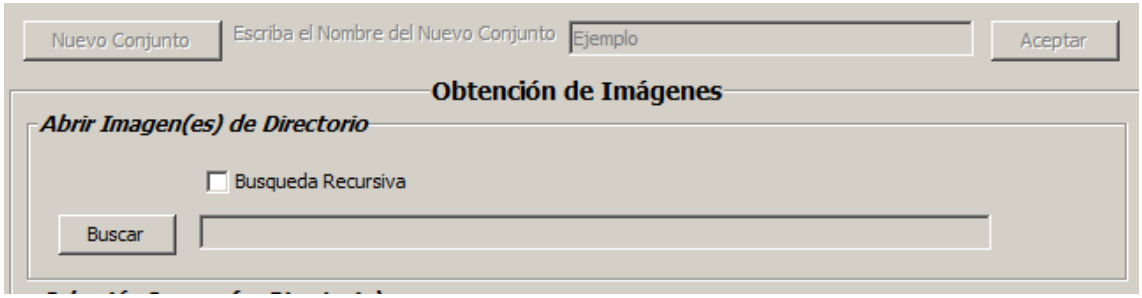

Figura 48: Cuadro de búsqueda de imágenes

Pulsar en *Buscar* para cargar imágenes desde el disco duro. Se abrirá una ventana de búsqueda, donde se seleccionará un directorio con imágenes. Se puede ver en la Figura 49:

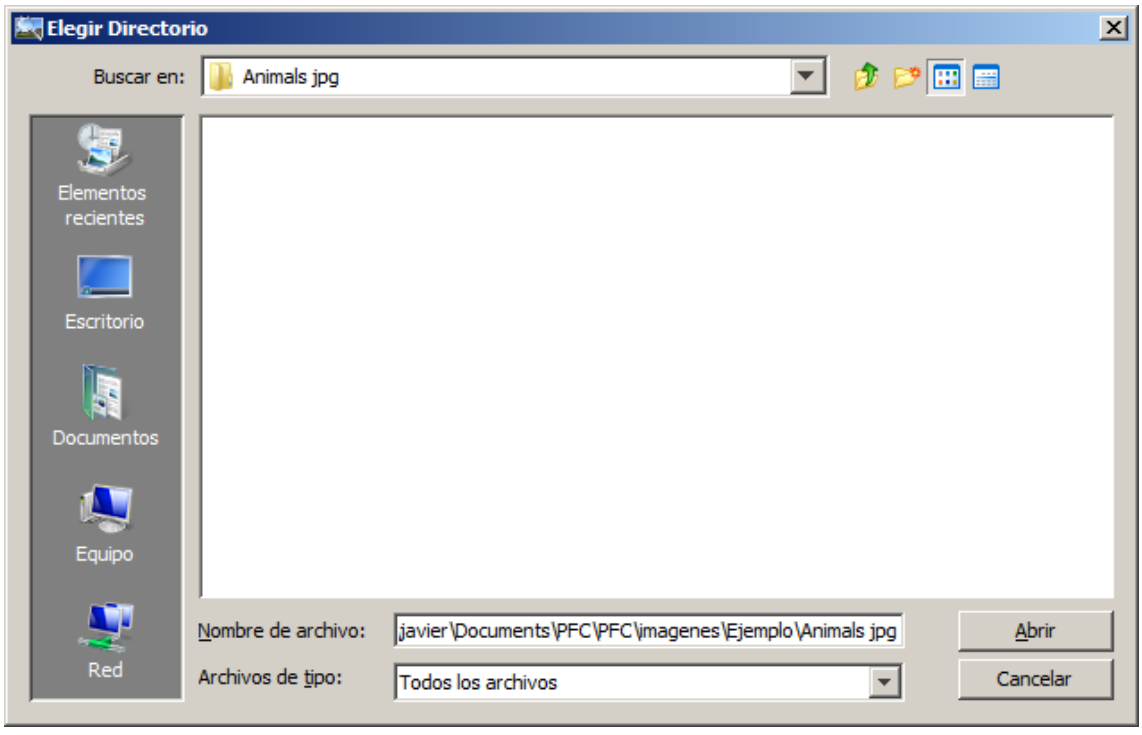

Figura 49: Cuadro para la elección del directorio de imágenes

No se puede seleccionar ningún fichero porque lo que se pide abrir es un directorio. Una vez abierto se cargarán todas las imágenes que hay en él. Si se elige *Búsqueda Recursiva* se abrirán

todas las imágenes de todos los subdirectorios del directorio elegido, de forma recursiva, hasta que no queden más imágenes.

Si el directorio contiene muchas imágenes, espere a que se carguen todas en la tabla. Una vez se han cargado todas las imágenes aparecerá habilitada la tabla de *Selección de Imágenes,* sus botones, y la zona de *Filtrado de las Imágenes Seleccionadas*.

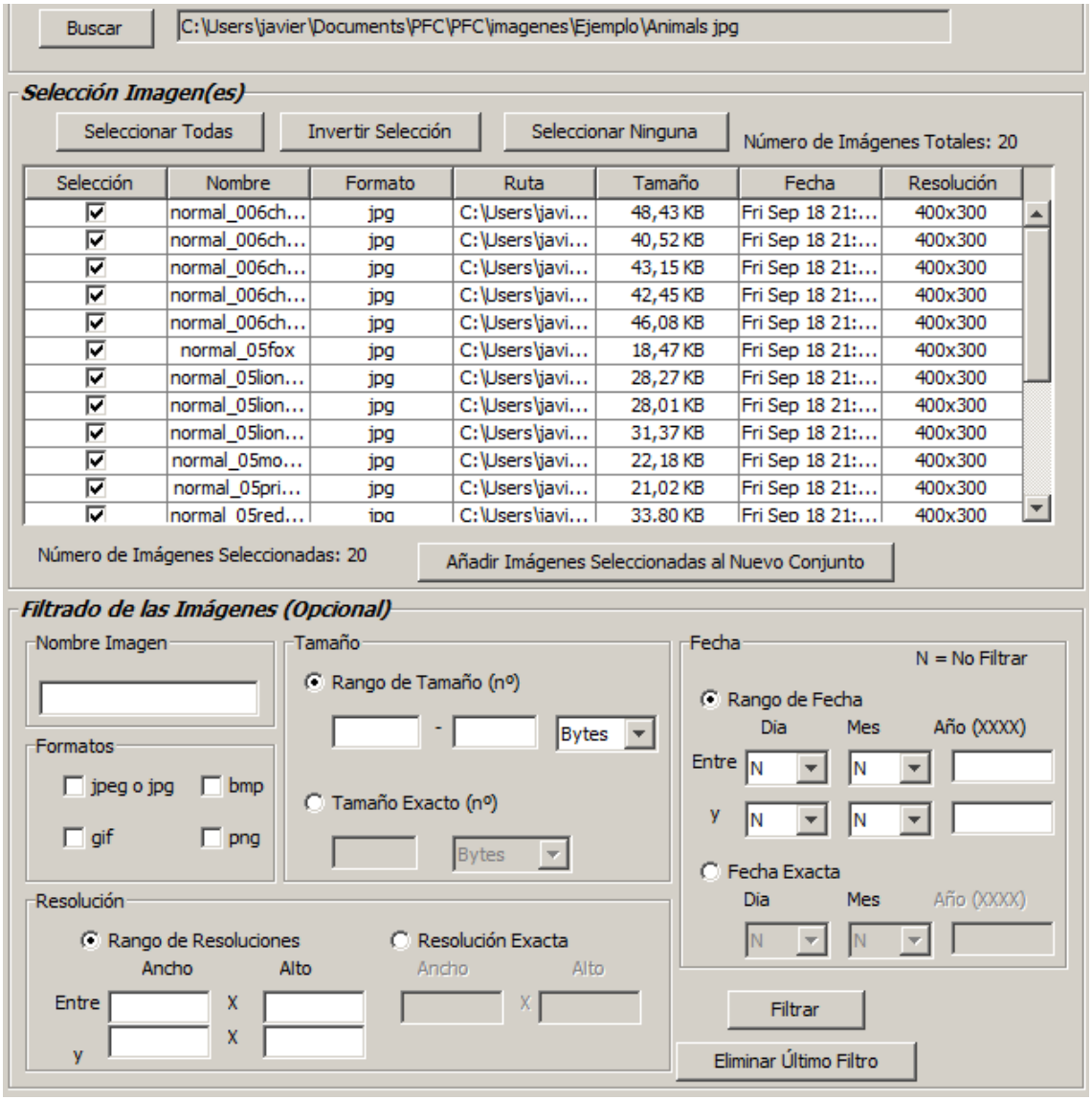

Figura 50: Cuadro de selección de imágenes

En la zona de *Selección de Imágenes,* Figura 50, se pueden seleccionar las imágenes que se desee, usando las filas de la columna *Selección* o usando los botones de *Selección.* Si estas opciones no son suficientes para seleccionar las imágenes deseadas, se pueden las diferentes opciones que incluye el filtrado. El *Filtrado* realiza una búsqueda sobre las imágenes de la tabla. Se puede buscar sobre ellas combinando todos los atributos de las imágenes como criterios de búsqueda. Cuando se ha terminado de elegir los criterios de búsqueda, pulsar en *Filtrar.* En la tabla se mostrarán las imágenes resultantes del filtrado ejecutado. Si el filtrado no fuese satisfactorio pulsar el botón *Eliminar Último Filtro*, el cual eliminará sólo el último filtro

realizado, es decir, si se encadenasen varios filtros, y se diese al botón, no se recuperarían las imágenes que se abrieron inicialmente desde el directorio.

4.2 Adición de Imágenes al Conjunto

Cuando se seleccionen las imágenes deseadas pulse en *Añadir Imágenes Seleccionadas al Nuevo Conjunto.* Esto añadirá las imágenes seleccionadas al conjunto que se esté creando. Pudiendo repetir este paso las veces que se consideren necesarias, si por ejemplo, se desea que el conjunto contenga imágenes de directorios distintos.

Si se pincha, con el ratón, en cualquier fila de las tablas, se cargará la imagen seleccionada, como se puede ver en la Figura 51:

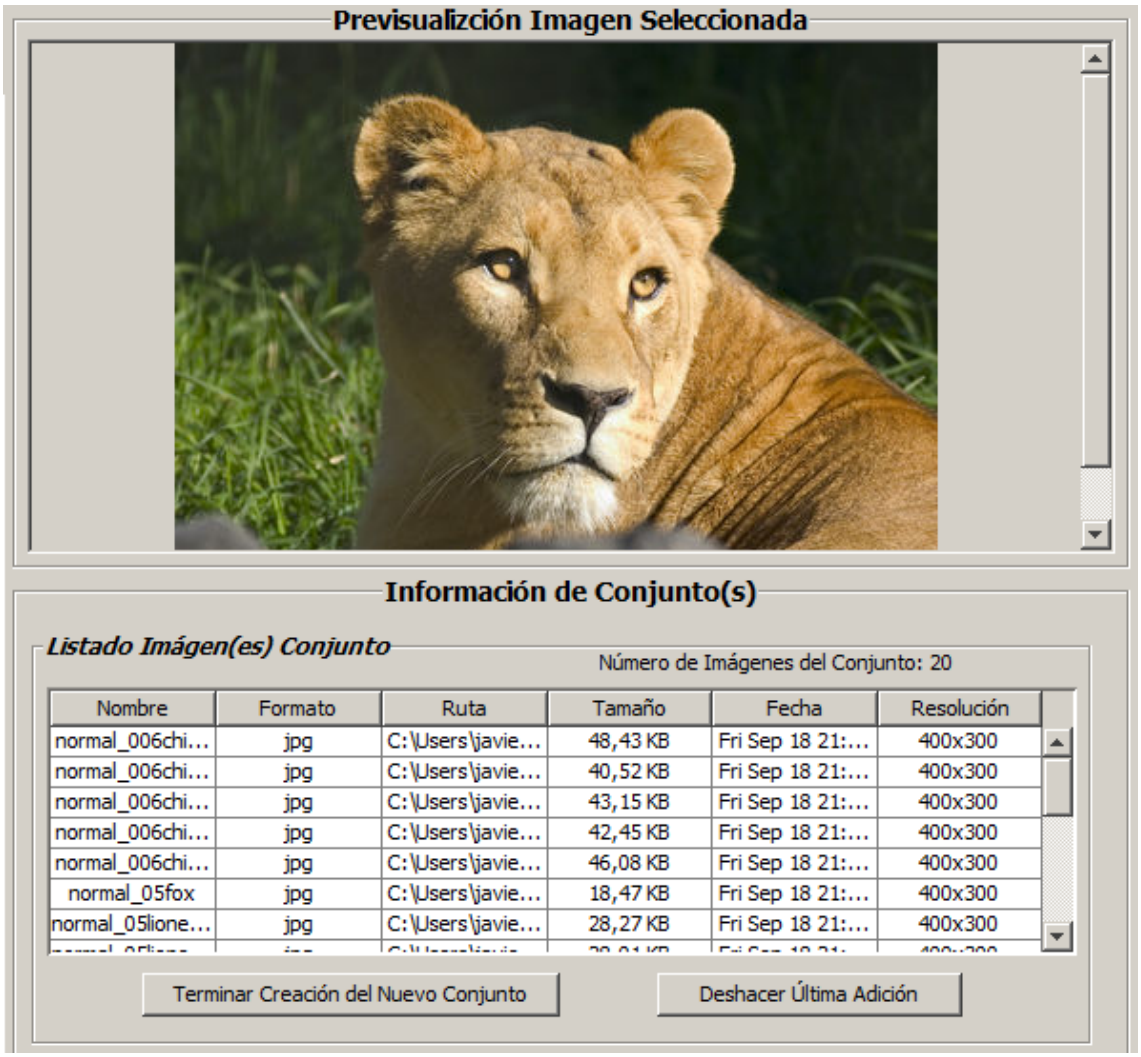

Figura 51: Cuadro con el listado de las imágenes del conjunto

En el *Listado Imagen(es) Conjunto*, Figura 51, se encuentran las imágenes que pertenecen al conjunto que se está creando. Pudiendo añadir más a este, como se ha explicado anteriormente. Si las últimas imágenes añadidas al conjunto se desean eliminar, se puede pulsar el botón *deshacer Última Adición,* para eliminar la ultima inserción de imágenes en el conjunto. Cuando el conjunto disponga de todas las imágenes deseadas, pulsar el botón *Terminar Creación del Nuevo Conjunto*.

4.3 Panel de Conjuntos Creados

Se habilitará la última zona aún deshabilitada de la ventana. Es la parte de los *Conjuntos Creados*. Aquí se listarán todos los conjuntos que se vayan creando.

Se puede crear otro nuevo conjunto dando al botón *Nuevo Conjunto* o trabajar con los conjuntos creados, mostrados en la Figura 52.

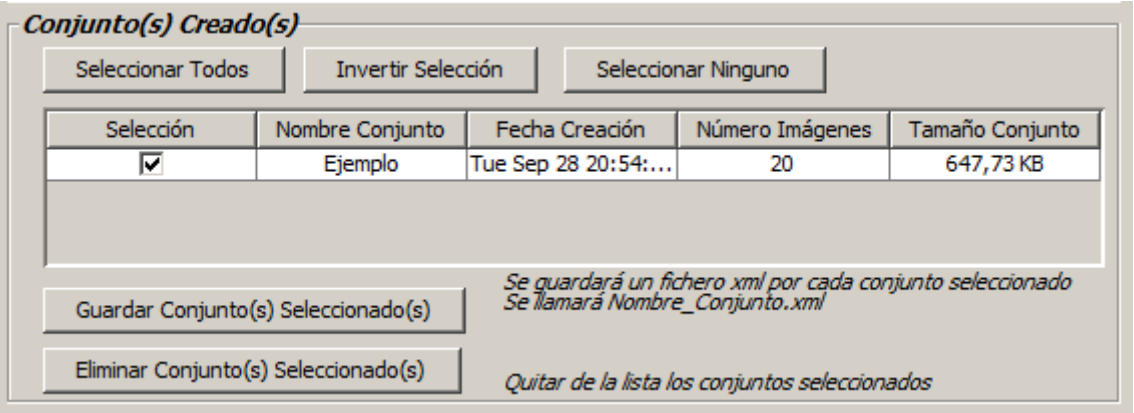

Figura 52: Panel de los conjuntos creados

Existen varias opciones de *Selección* de los conjuntos creados. Esta selección sirve para poder eliminar los conjuntos seleccionados y/o para guardar los conjuntos seleccionados. Si se pulsa *Eliminar Conjunto(s) Seleccionado(s)* se eliminarán los conjuntos que estén marcados en su columna de *Selección*, desapareciendo de la lista. Si se pulsa en *Guardar Conjunto(s) Seleccionados* se guardarán todos los conjuntos seleccionados en el directorio donde se elija, ya que se abrirá una ventana de guardado, como la mostrada en la Figura 53.

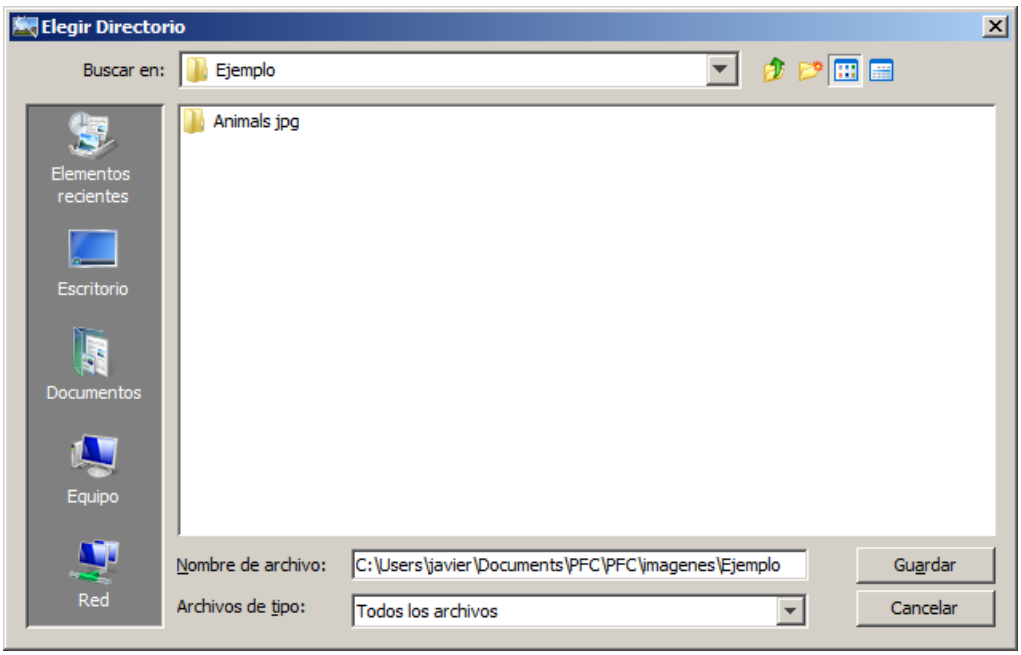

Figura 53: Ventana de elección del directorio donde guardar el conjunto

Se guardarán los conjuntos seleccionados en ficheros *XML*. Destacar que *estos ficheros tienen referencias a las imágenes en una ubicación absoluta, por lo tanto, el fichero XML sólo servirá para utilizar con las imágenes mientras estas estén en la misma ubicación*.

Al acabar, cerrar la ventana. Cuando se cierre una ventana de JUBSAC, si en esta se podían guardar datos, se avisará siempre antes de cerrar, por si no terminó de guardar los datos con los que trabajaba. En la Figura 54, se muestra el aviso:

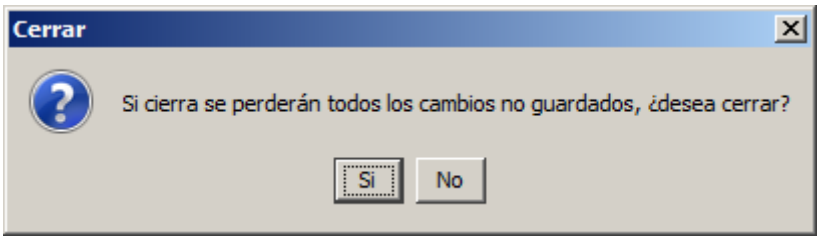

Figura 54: Ventana de aviso de cerrar

5. Conversor de Imágenes

Si en el menú principal se pulsa en *Convertir Formatos Imágenes* se abrirá una ventana como la mostrada en la Figura 55:

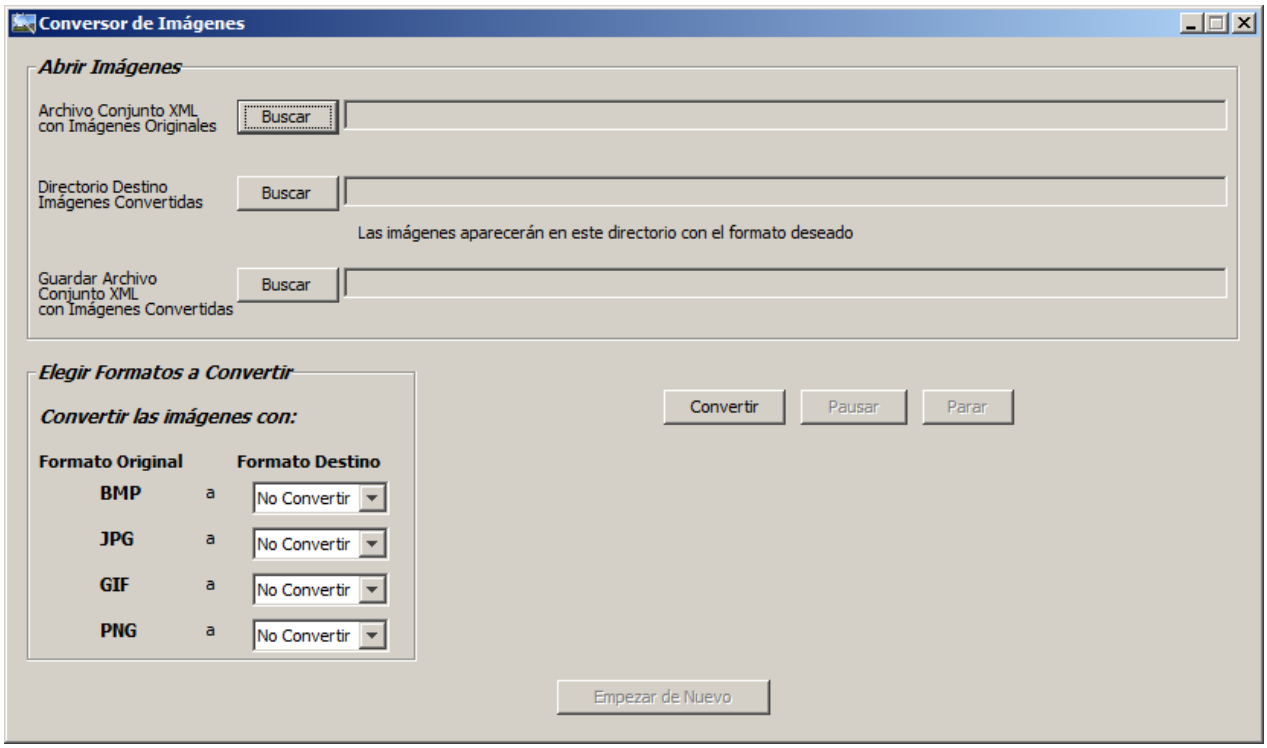

Figura 55: Ventana del conversor de formatos de imágenes

Se tendrá que elegir el fichero con formato *XML* con el conjunto de las imágenes originales que se desea convertir. Se elegirá el ejemplo creado, en la Figura 56, para convertirlas a GIF

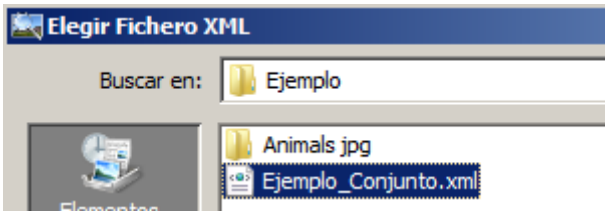

Figura 56: Abrir XML para conversión

También es necesario especificar el directorio de destino donde se guardarán, las imágenes convertidas y el nuevo archivo *XML* con el conjunto de las imágenes convertidas. Por último, Figura 59, hay que elegir los formatos originales que se desean convertir a que formatos destino.

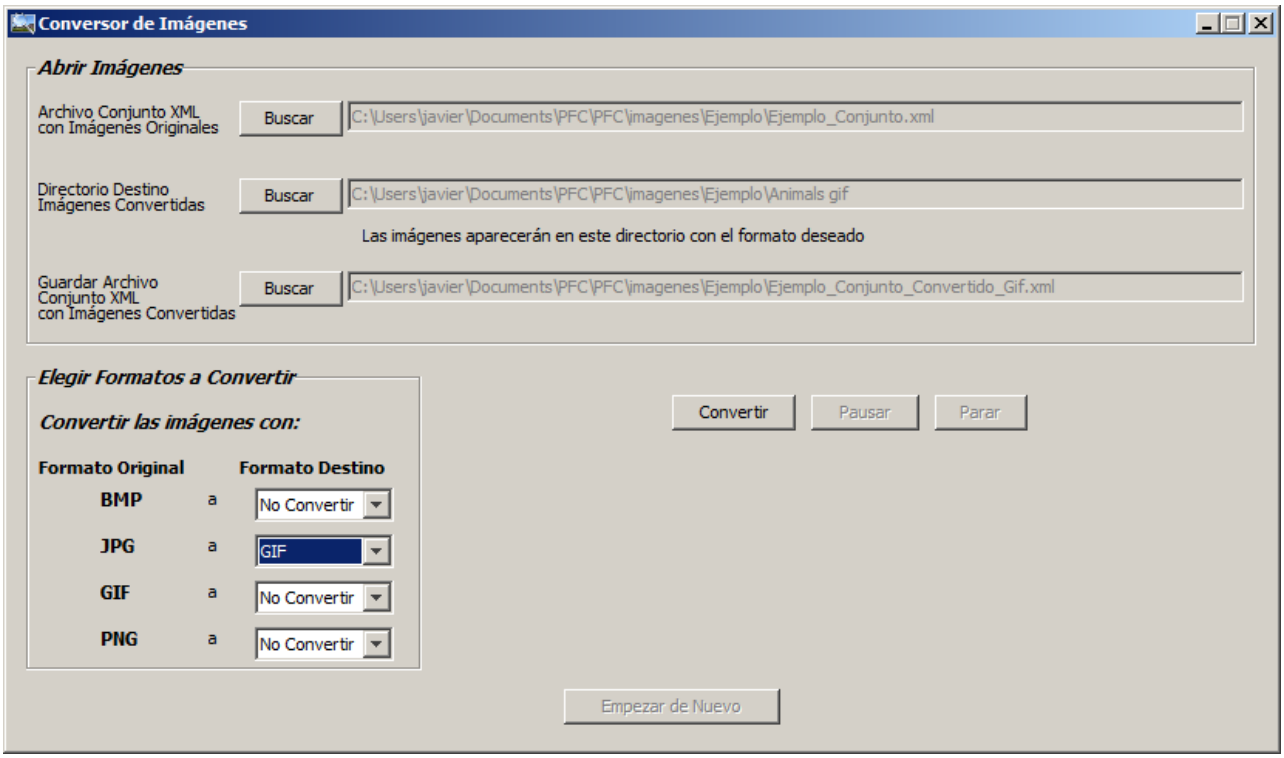

Figura 57: Ventana del conversor de formatos de imágenes configurado

Una vez se pulse *Convertir*, comenzará la conversión, como se muestra en la Figura 58:

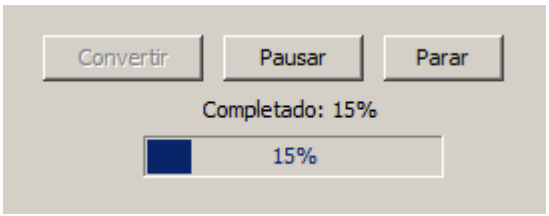

Figura 58: Progreso de conversión en curso

Se puede *Pausar* esta conversión, para después poder continuarla, dándole a *Convertir*, o para pararla definitivamente, dando a *Parar*. Si se pausa y después se continúa, el tiempo total se verá afectado. Si para la conversión, en el *XML* sólo estarán las imágenes que se hayan convertido antes de la detención.

Al acabar, se guardará el Nuevo Conjunto. Esta tarea puede llevar algún tiempo si el conjunto es muy grande. Y cuando finalice del todo se mostrará el tiempo total, Figura 59:

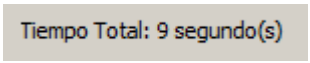

Figura 59: Tiempo total conversión

Después pulsar *Empezar de Nuevo,* para realizar un proceso nuevo de conversión o pulsar el botón de cerrar para terminar con la herramienta de conversión.

6. Conversión de Imágenes al Dominio Transformado

Este complemento permite transformar conjuntos de imágenes usando la transformada del Coseno, de Fourier, Ondulada, y sus inversas. La Figura 60 muestra la ventana de *Convertir Conjuntos al Dominio Transformado*:

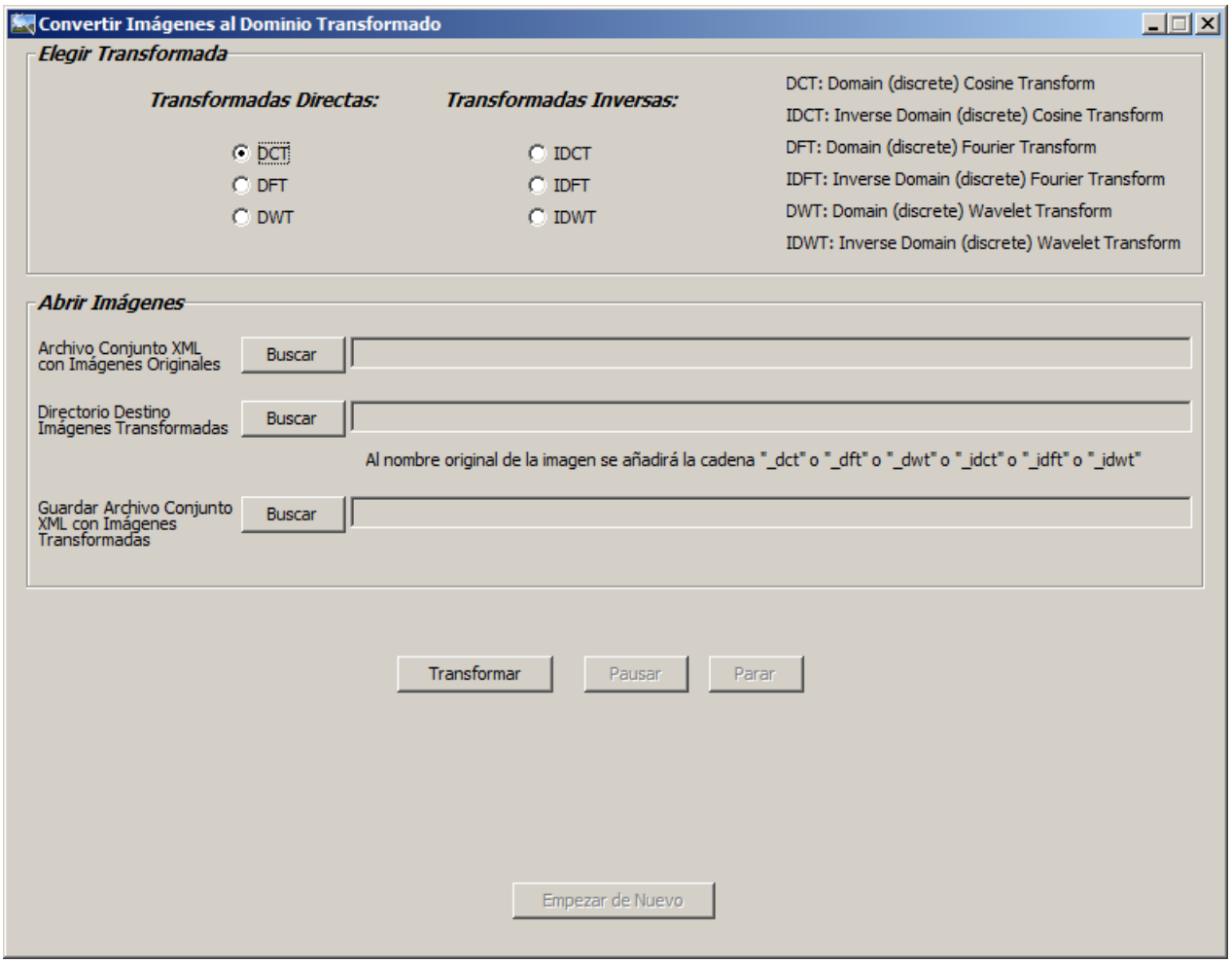

Figura 60: Convertir conjuntos al dominio transformado

Primero hay que elegir que transformada se quiere realizar (DCT, DFT, DWT, IDCT, IDFT, IDWT). Después elegir el conjunto *XML* con las imágenes a transformar (estas pueden estar en cualquier formato). También el directorio de destino donde se colocaran las imágenes transformadas. A las imágenes transformadas se les añadirá al final del nombre y antes del formato, una terminación (_dct, _idct, etc.) para indicar el tipo de transformada que se ha aplicado. Ver proceso en Figura 61:

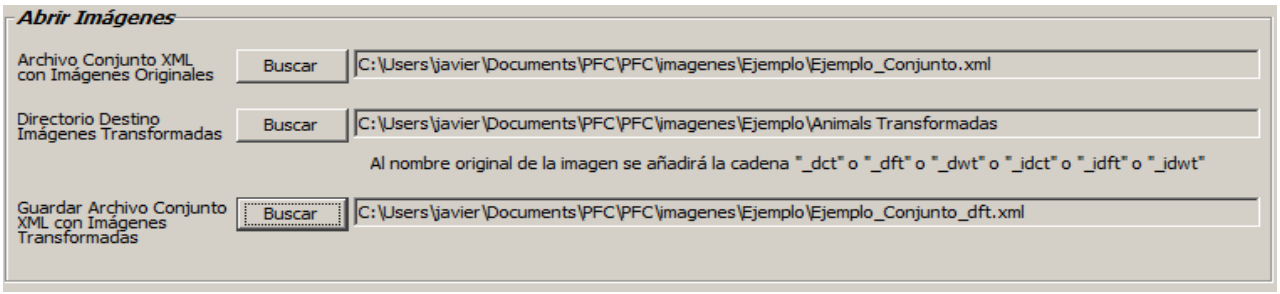

Figura 61: Abrir imágenes para transformación

Después pulsar en *Transformar* y comenzará la transformación, como se observa en la Figura 62:

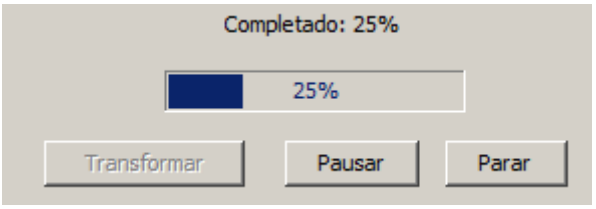

Figura 62: Comenzar transformación

Se podrá *Pausar* esta transformación, Figura 63, para después poder continuarla, pulsando en *Transformar*, o para pararla definitivamente, pulsando *Parar*. Si se pausa y después continua el tiempo total se verá afectado.

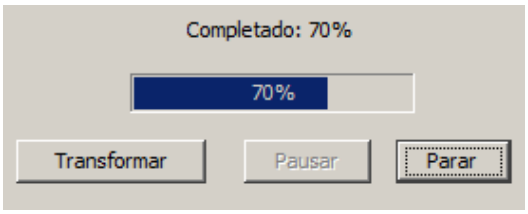

Figura 63: Pausar transformación

Al acabar, se guardará el Nuevo Conjunto, Figura 64. Esta tarea puede llevar algún tiempo si el conjunto es muy grande.

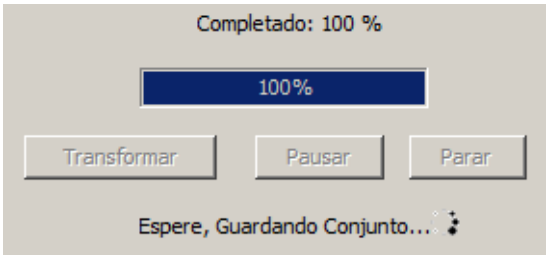

Figura 64: Guardando conjunto resultado de la transformación

Y cuando finalice del todo se mostrará el tiempo total, Figura 65:

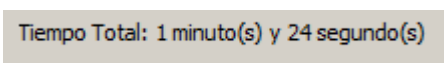

Figura 65: Tiempo total de la transformación

Si se para la transformación, en el *XML* sólo estarán las imágenes que se hayan transformado antes de la detención. Después se podrá pulsar *Empezar De Nuevo,* Figura 66, para realizar un proceso nuevo de transformación o cerrar la herramienta de transformadas.

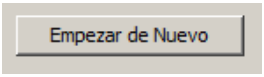

Figura 66: Empezar de nuevo la transformación

Dar al botón de cerrar de la ventana para terminar con la herramienta de transformación.

7. Aplicar Técnicas Esteganográficas a Conjuntos

Este módulo de la herramienta permite ejecutar una batería de programas esteganográficos para ocultar imágenes en conjuntos.

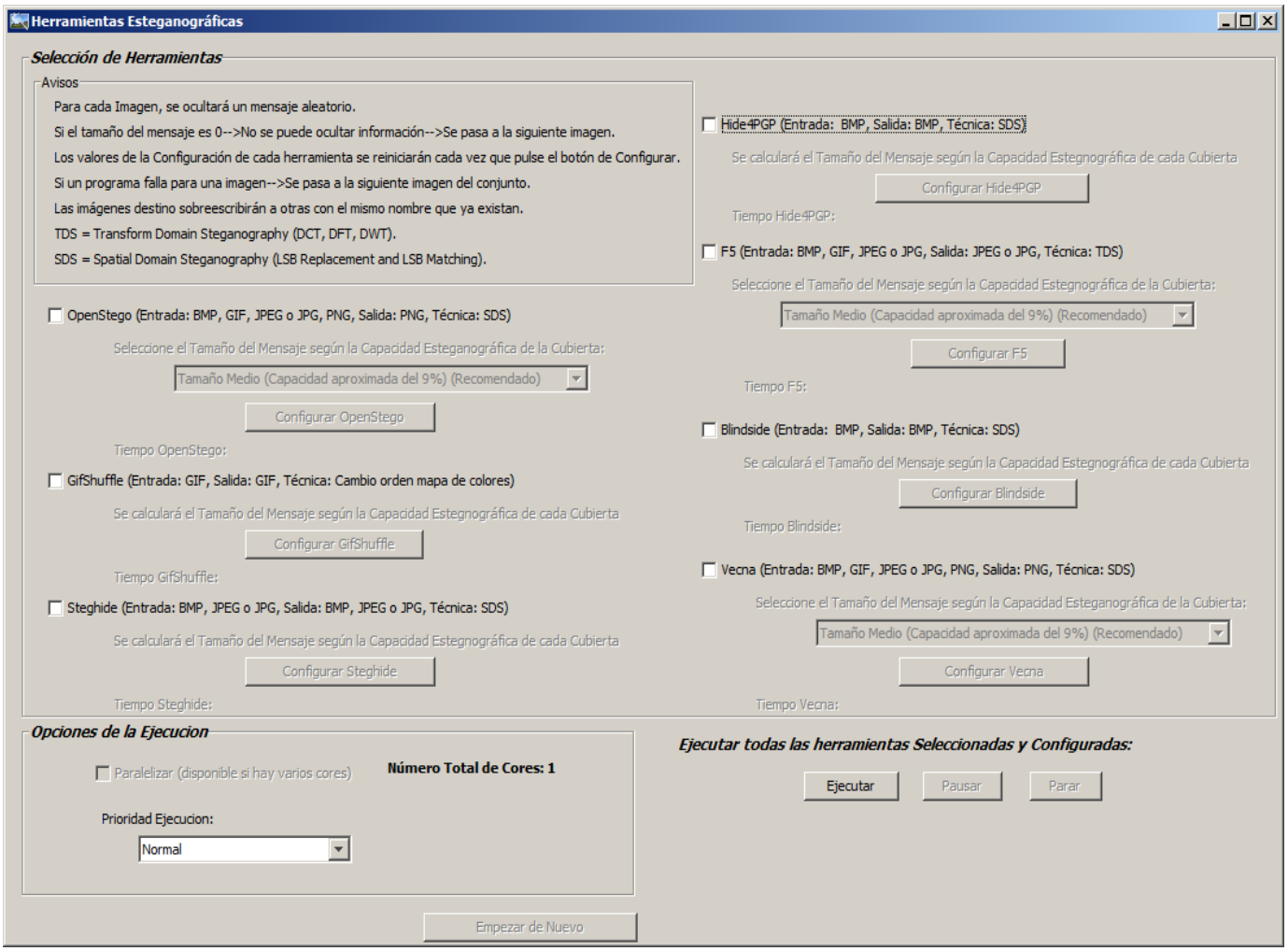

Figura 67: Ventana de aplicar técnicas esteganográficas a conjuntos

En la parte superior izquierda de la ventana, Figura 67, se dan una serie de avisos y notas, para que el usuario tenga unas premisas en cuenta sobre la ejecución de las herramientas.

Hay varias herramientas esteganográficas que seleccionar. El usuario puede seleccionar las que le parezcan oportunas. Después de seleccionar cada herramienta, se habilitará el botón de *Configurar*. El cual abrirá una nueva ventana, para cada herramienta. Hay dos tipos de herramientas: unas necesitan elegir el tamaño del mensaje a ocultar, y otras lo elijen ellas mismas. El tamaño que se elije es un porcentaje sobre el tamaño de la imagen que se oculte en cada momento.

7.1 Configuración Específica

Para cada herramienta se da información sobre su nombre, sus formatos de entrada, sus formatos de salida, y el tipo de técnica estegoanalítica usada.

Si se selecciona por ejemplo *OpenStego,* Figura 68:

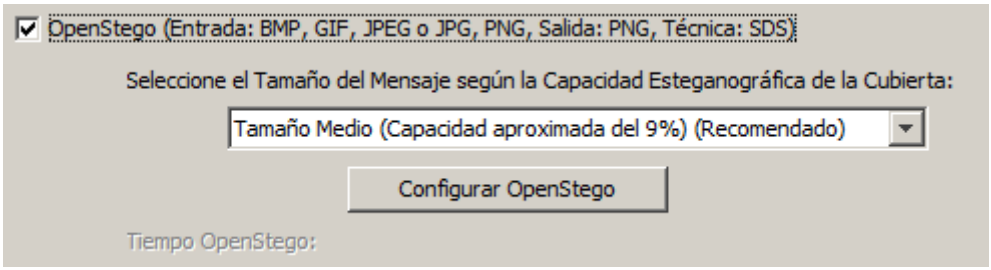

Figura 68: Seleccionar OpenStego

Si se pulsa el botón de *Configurar OpenStego* se muestra una ventana como la de la Figura 79:

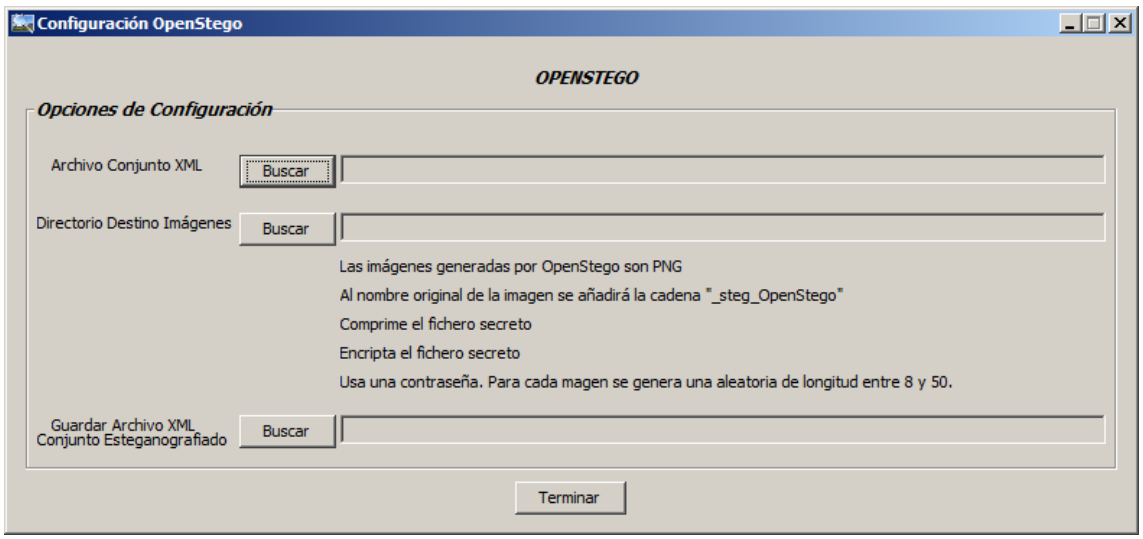

Figura 69: Configuración de OpenStego

Por cada herramienta esteganográfica que se configure, habrá una ventana nueva como la anterior. En esta ventana habrá que abrir el conjunto con las imágenes en las que se quiere ocultar información. También se abrirá el directorio de destino, donde se guardarán las imágenes con información oculta, usando la herramienta elegida. Y por último, se abrirá el archivo del conjunto *XML* que hace referencia a las imágenes esteganografiadas. Al acabar pulsar el botón de *Terminar* y si se quiere cerrar la ventana, sin mantener la configuración dar al botón de cerrar la ventana.

7.2 Configuración General

De vuelta ya, en el menú anterior, se puede elegir entre unas *Opciones de la Ejecución*. Se puede paralelizar, si se dispone de más de un núcleo de procesamiento. También se puede elegir la prioridad de la ejecución, esto dará más o menos recursos de procesamiento a la ejecución.

Por último, pulsar el botón de *Ejecutar.* Este ejecutará todas las herramientas esteganográficas elegidas, una detrás de otra, ocultando información en cada imagen de forma paralela. Cuando acabe, se guardará el conjunto de la última herramienta ejecutada y se podrá empezar una nueva ejecución pulsando en *Empezar de Nuevo.*

8. Extracción de Medidas de Imágenes

Este módulo sirve para extraer información de imágenes que luego serán utilizadas como atributos en las instancias de los algoritmos de aprendizaje automático de la fase de experimentación. La interfaz se muestra en la Figura 70:

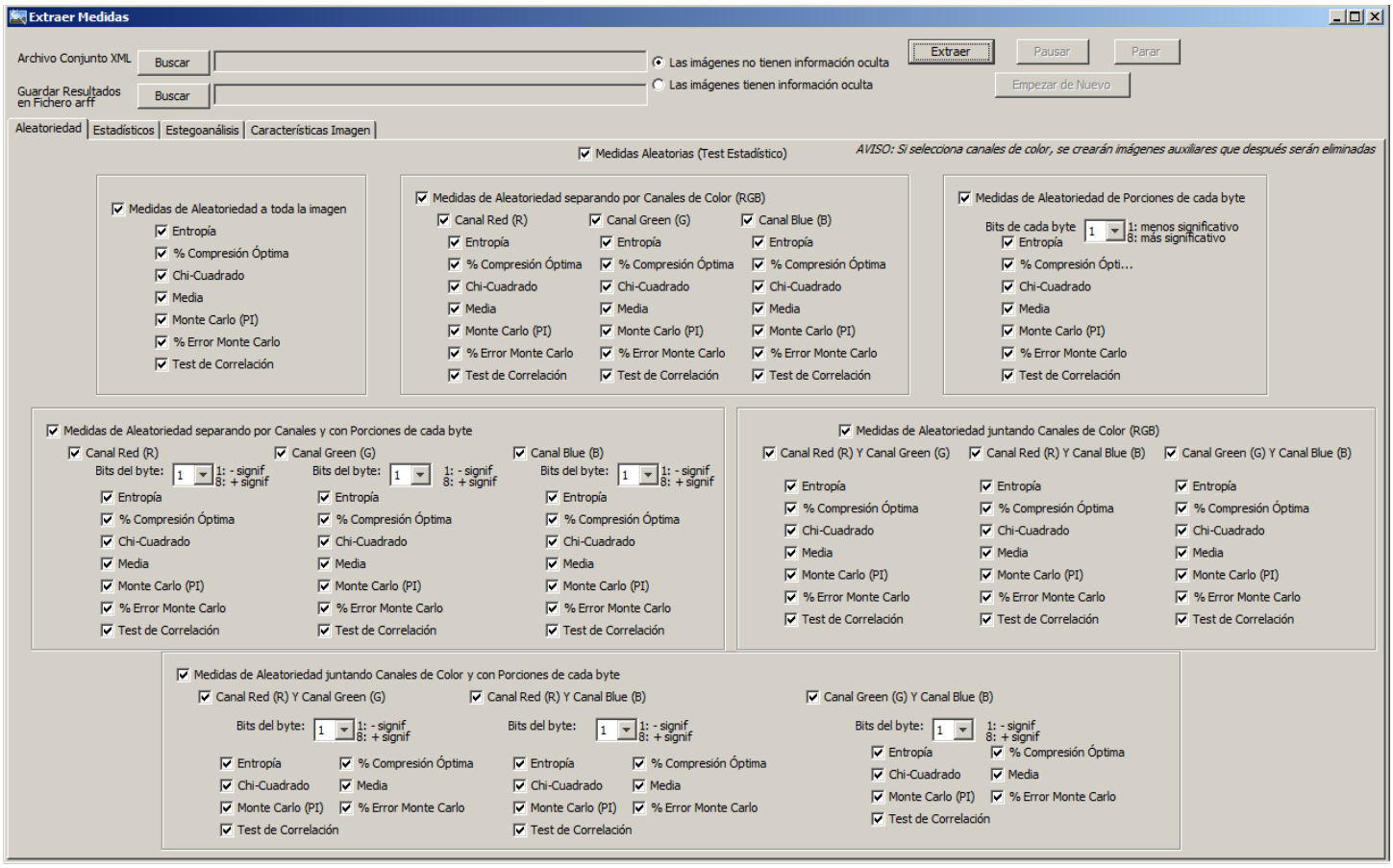

Figura 70: Ventana de la extracción de medidas de imágenes

El usuario debe realizar tres pasos, en cualquier orden, eso sí, siempre antes de la ejecución: selección de los ficheros, selección de la clasificación de las instancias y selección de las medidas a extraer.

8.1 Selección de Ficheros

En primer lugar el usuario debería seleccionar el conjunto *XML* de imágenes de las cuales quiere extraer la información que seleccionará después. A continuación, se elegirá el nombre y la ubicación del fichero *arff* de Weka, donde se escribirán los patrones.

8.2 Selección de la Clasificación de las Instancias

Después seleccionar si las imágenes tienen información oculta o no. Esto sirve para dar una clase a las instancias que se crearán. Todas tendrán la misma, por lo que si se quiere tener en un mismo fichero de instancias (*arff*) con distintas clases se tendrá que usar la herramienta de fusionado de ficheros de instancias.

En la Figura 71 se pueden ver los dos pasos anteriores ya aplicados.

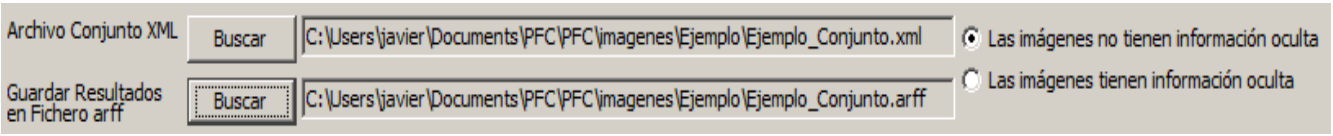

Figura 71: Configuración de la extracción de medidas de imágenes

8.3 Selección de las Medidas a Extraer

El usuario seleccionará también que medidas quiere extraer. Estas están divididas en cuatro pestañas:

- **Medidas de la Aleatoriedad**: Son test estadísticos para medir la aleatoriedad sobre secuencias de bytes. Es usado para generadores de números pseudoaletorios.
- **Medidas Estadísticas**: Son medidas estadísticas de primer orden para medir la dispersión sobre conjuntos de datos. En este caso los datos usados son los píxeles de las imágenes.
- **Medidas Estegoanalíticas**: Son medidas que se calculan en técnicas estegoanalíticas creadas por Jessica Fridrich. Sirven, sobre todo, para romper técnicas de LSB.
- **Características de la Imagen**: Son medidas muy básicas que se encuentran en todas las imágenes y que dan información sobre ella.

Cuando se hayan elegido todas las medidas deseadas, pulsar el botón *Extraer*. El proceso se podrá pausar y parar, pero estos no serán instantáneos, sino que se tendrá que terminar la imagen que está en proceso, lo que llevará apenas unos segundos, Figura 72.

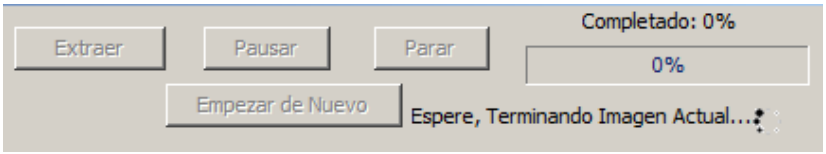

Figura 72: Pausar extraer medidas de imágenes

Si se para la extracción, en el fichero de instancias destino sólo se escribirán las instancias que se hayan llevado hasta el momento de la detención.

9. Fusión de Ficheros de Patrones

Esta parte es un complemento, para unir varios ficheros de instancias (*arff*), con los mismos atributos, en un solo fichero. Es necesario, porque en la extracción de medidas sólo se puede elegir una clase por fichero de instancias (*arff*). Entonces se conseguirá tener un fichero de instancias con varias series de instancias con varias clases. Además a esto se le une el hecho de que en la fase de experimentación se podrán aleatorizar estas instancias antes de entrenar.

El usuario dará al primer botón de *Buscar* para ir añadiendo ficheros de instancias a la lista. En la lista se mostrará información relevante de cada fichero añadido, como su nombre o el número de patrones que tiene. También deberá elegir el fichero de destino, pulsando en el segundo botón *Buscar*. La interfaz al comienzo es la mostrada en la Figura 73:

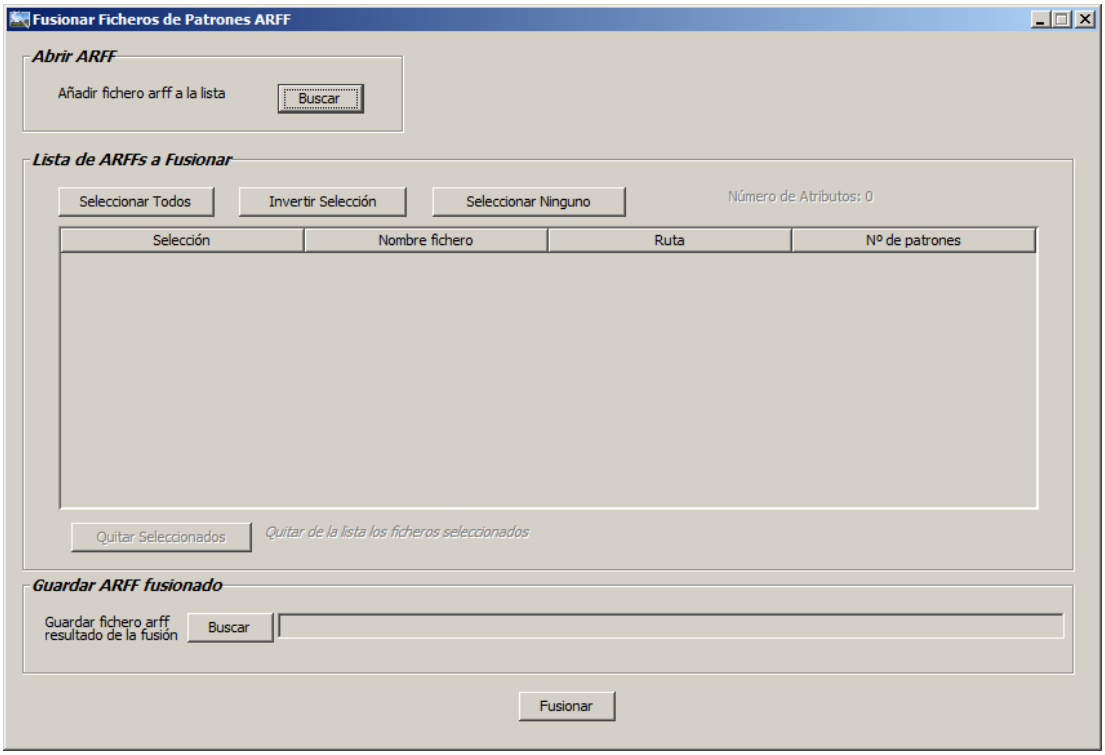

Figura 73: Ventana de la fusión de ficheros de patrones ARFF

Si se añaden varios ficheros y se completan todos los campos se tiene la Figura 74:

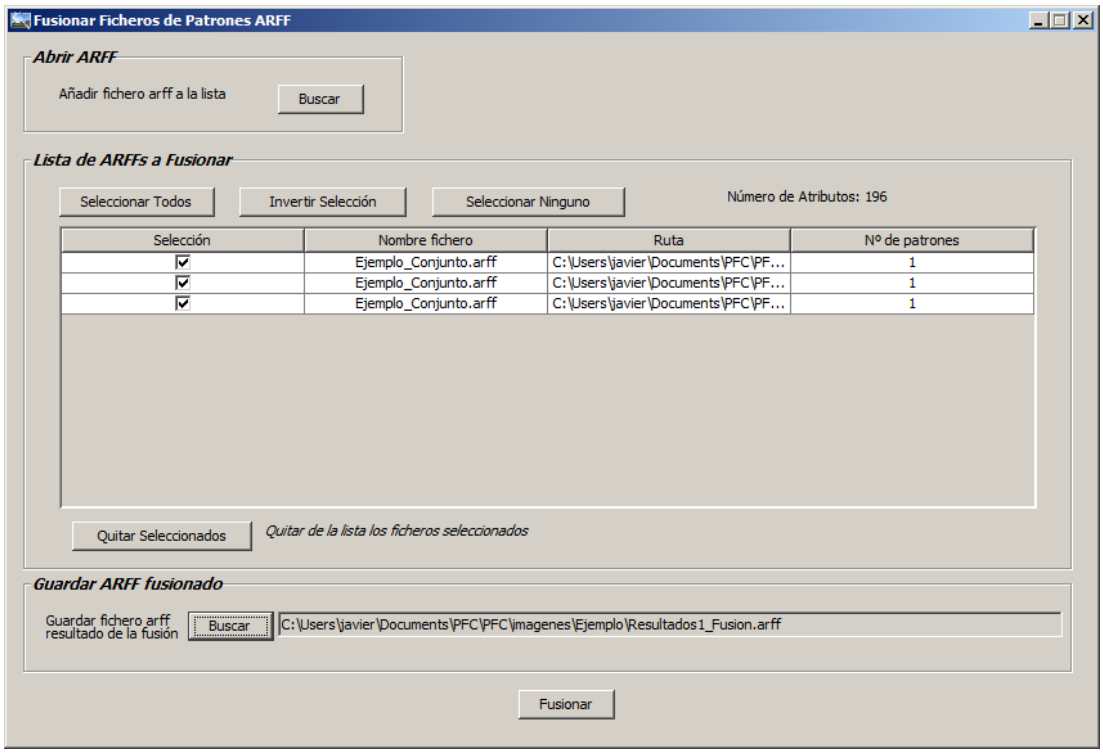

Figura 74: Ventana de la fusión de ficheros de patrones ARFF configurada

Finalmente pulsar en *Fusionar* y se guardará el fichero con todos los patrones seleccionados de la lista. Los patrones no seleccionados no serán añadidos.

10. Creación de Experimentos

Este módulo permite la creación de diferentes experimentos utilizando los algoritmos de aprendizaje automático. La interfaz se muestra en la Figura 75:

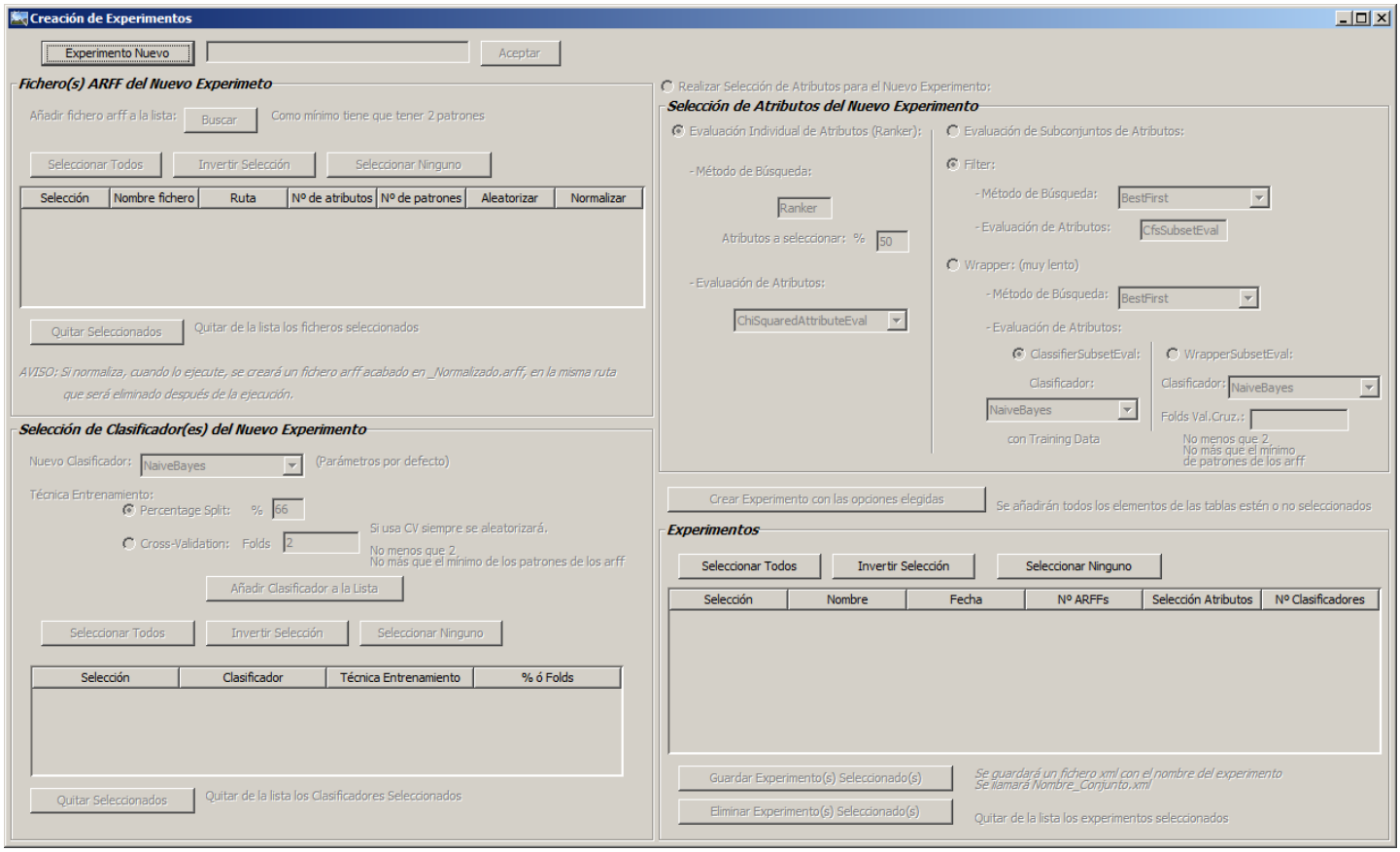

Figura 75: Ventana de la creación de experimentos

Estos experimentos, se guardarán en ficheros *XML*, para poder cargarlos después. Un experimento está formado por uno o varios ficheros de instancias (*arff*), una o ninguna selección de atributos y uno o varios clasificadores. Empezar pulsando sobre el botón de *Experimento Nuevo* y escribir el nombre del experimento, como se ve en la Figura 76:

Figura 76: Escritura del nombre del experimento nuevo

10.1 Selección de Ficheros de Instancias

Una vez escrito el nombre del nuevo experimento, se habilitarán varias zonas de la pantalla, Figura 77, entre ellas la parte de los Ficheros Arff:

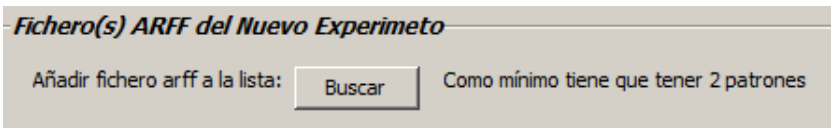

Figura 77: Botón de buscar ARFF

Donde se buscará un nuevo fichero de instancias (*arff*) para el experimento. Según se busquen, se irán añadiendo estos a la tabla de la Figura 78:

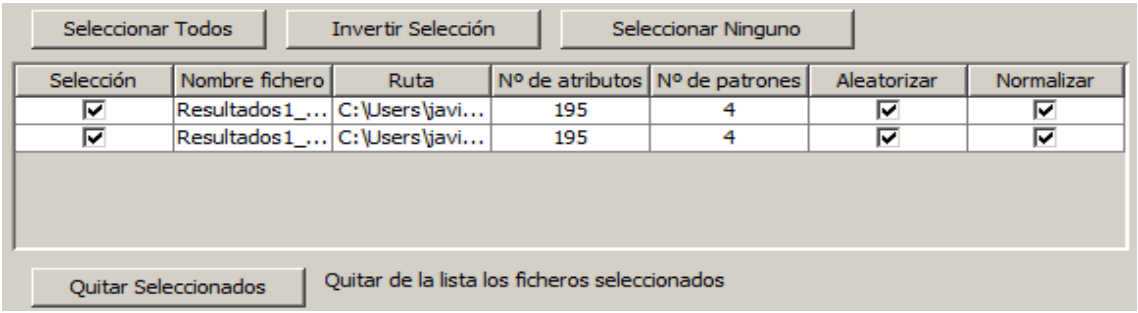

Figura 78: Lista de ficheros ARFF del experimento

Se pueden seleccionar, y después eliminar de la tabla los seleccionados. También se puede elegir si se quiere que las instancias se **aleatoricen** y/o se **normalicen**, en el momento de la ejecución del experimento, aunque si se ejecuta Validación cruzada (Cross Validation) siempre se aleatorizará, ya que esta técnica siempre aleatoriza. La normalización, significa, que cada atributo se sustituirá por otro con un valor entre 0 y 1, guardando la proporción.

10.2 Selección de Atributos

Una vez elegidos los ficheros de instancias, se puede elegir usar selección de atributos sobre los ficheros elegidos. Para ello se marcará el botón de la Figura 79:

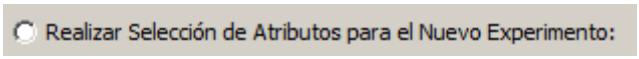

Figura 79: Botón de realizar selección de atributos

Si el usuario desea que se realice selección de atributos, si no, dejarla desmarcada. Si se marca, se puede configurar la selección de atributos según se desee (Figura 80), eligiendo los métodos de búsqueda y de evaluación, así como otros parámetros que pueda necesitar.

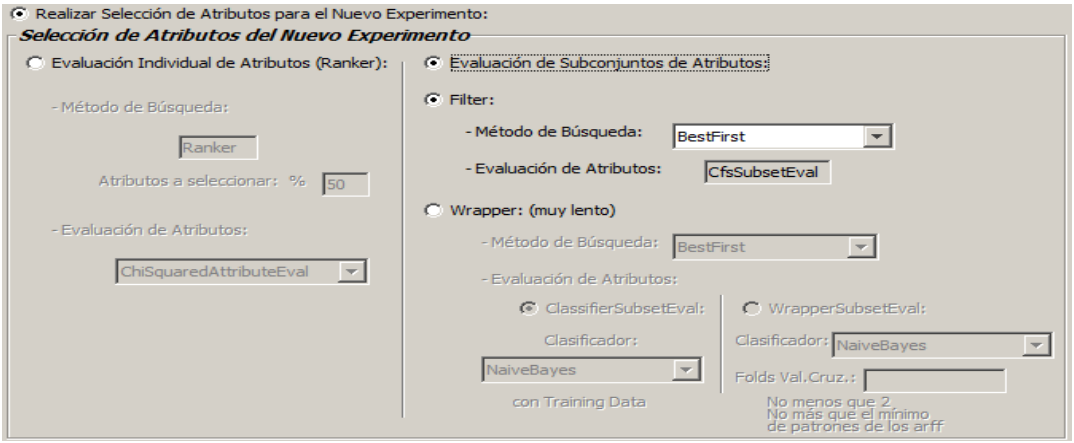

Figura 80: Zona de la selección de atributos

10.3 Selección de Clasificadores

A continuación, el usuario debe ir a *Selección de Clasificadores***,** donde se elegirá que clasificador añadir a la lista, y sus parámetros, como se muestra en la Figura 81:

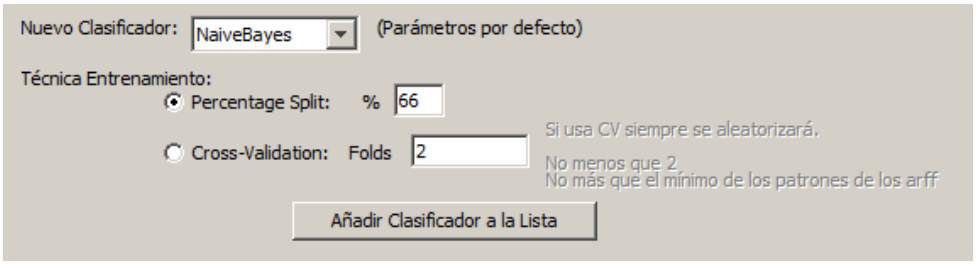

Figura 81: Zona de añadir clasificador

Pulsar *Añadir Clasificador a la Lista.* Se pueden añadir todos los clasificadores que estime oportuno. Según se vayan añadiendo irán apareciendo en la tabla, Figura 82:

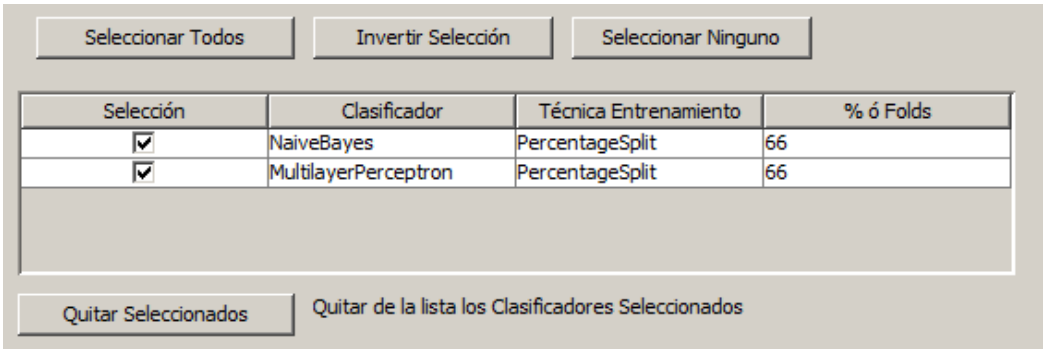

Figura 82: Lista de clasificadores

Cuando se acabe de seleccionar los ficheros de instancias, la selección de atributos y los clasificadores pulsar el botón *Crear Experimento con las opciones elegidas,* Figura 83.

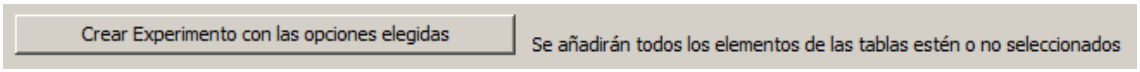

Figura 83: Botón de creación del nuevo experimento

10.4 Selección de Clasificadores

Después de pulsar sobre el botón de la figura 83, se habilitará la zona de experimentos, Figura 84:

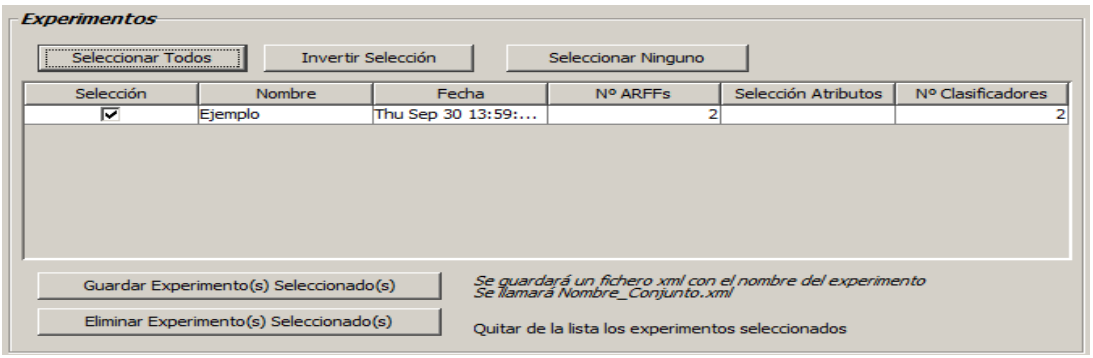

Figura 84: Lista de experimentos creados

Para crear más experimentos bastará con repetir los pasos previos, o finalizar y dar al botón de *Guardar Experimentos Seleccionados* donde se elegirá el directorio donde guardar el *XML* con los datos del experimento.

11. Ejecución de Experimentos para Entrenamiento

En esta parte de la aplicación, se ejecutarán los experimentos *XML* creados con la *Creación de Experimentos*. Se ejecutarán cada uno de los clasificadores elegidos en el experimento para cada fichero de instancias (*arff*). Para cada clasificador entrenado se obtendrá un modelo en un fichero en formato *model* y un fichero TXT con los resultados del entrenamiento. La ventana se muestra en la Figura 85:

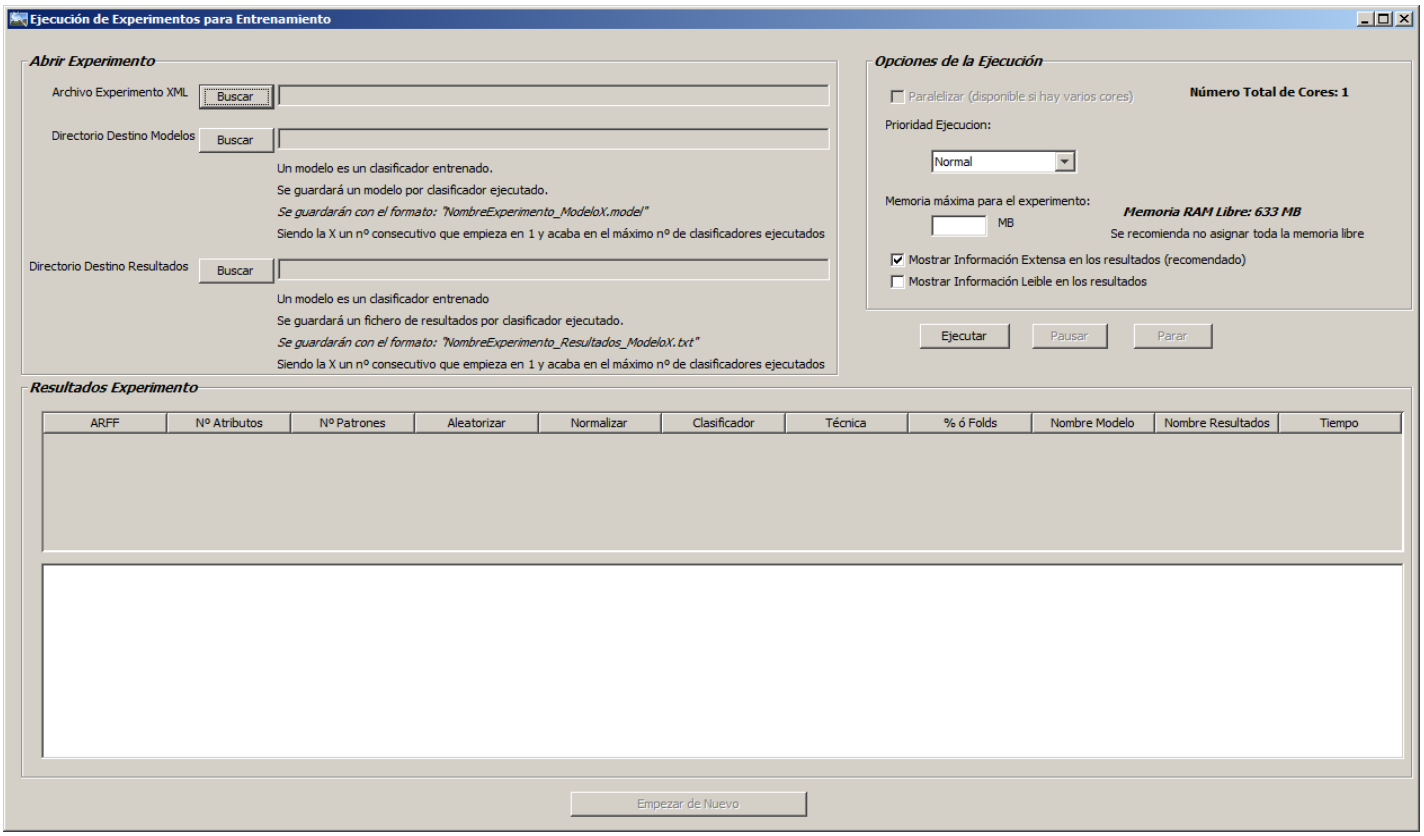

Figura 85: Ventana de ejecución de experimentos

El primer paso es abrir el fichero que contiene el experimento (en formato *XML*) que se quiere ejecutar, elegir un directorio donde guardar los modelos, y otro directorio donde guardar los resultados, que también se mostrarán por pantalla. También se podrán configurar varias opciones durante la ejecución, como paralelizarla (si dispone de varios núcleos), dar una prioridad a la ejecución de entre el resto de procesos del sistema operativo, dar una memoria máxima a cada clasificador, mostrar información de forma extensa y leible. Este primer paso es mostrado en la Figura 86:

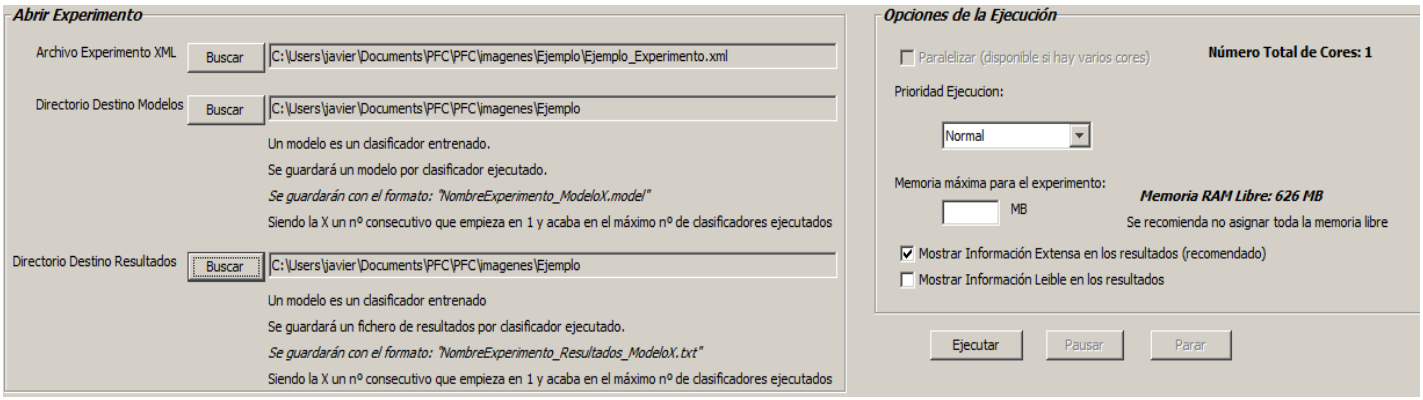

Figura 86: Configuración de la ejecución de experimentos

Después pulsar el botón de *Ejecutar*, para comenzar la ejecución. Se podrá pausar y luego continuar, o parar definitivamente.

Cuando acabe el entrenamiento se mostrará en una tabla los ficheros de instancias ejecutados, mostrando características relevantes de su ejecución, como el nombre de los ficheros donde se encuentra el modelo y los resultados. Si se pulsa sobre cualquier fila de la tabla, se mostrarán debajo los resultados, como se ve en la Figura 87:

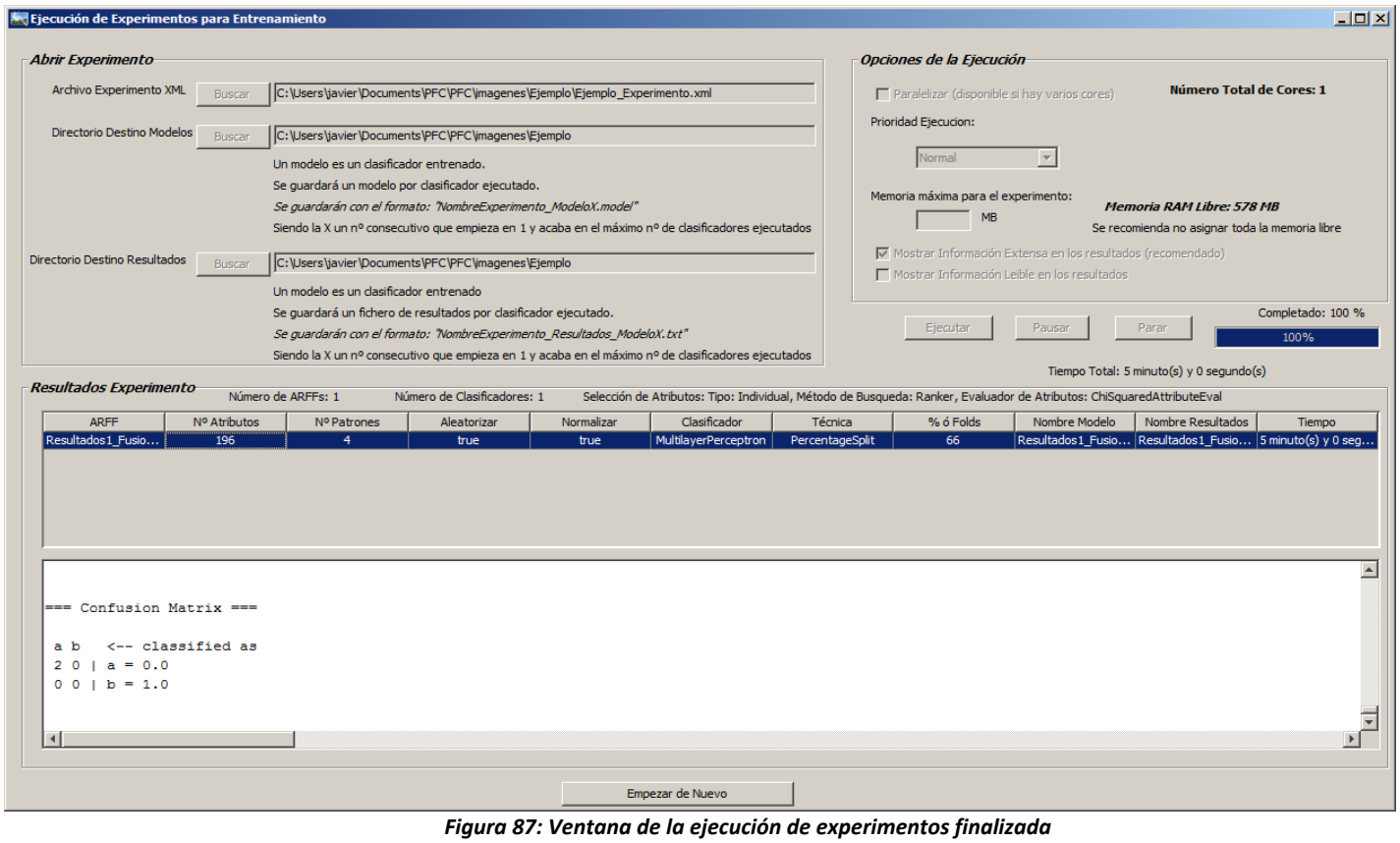

Figura 87: Ventana de la ejecución de experimentos finalizada

Una vez acabada la ejecución se podrá realizar otra nueva pulsando el botón *Empezar de Nuevo,* o se puede salir, pulsando el botón de cerrar de la ventana.

12. Ejecución de Modelos para Test

En este módulo se ejecutarán los modelos creados en la ejecución de experimentos para entrenamiento, para evaluar un nuevo conjunto de instancias y ver como son estas clasificadas. Si por ejemplo se dispone de un conjunto de imágenes que se quiere probar con un modelo, se creará previamente un conjunto de estas, se extraerán las mismas medidas que tuvieron las instancias con las que se ejecutó el modelo y se podrá ejecutar para ver como clasifica las nuevas instancias (0 si no tiene información oculta y 1 si tiene información oculta).

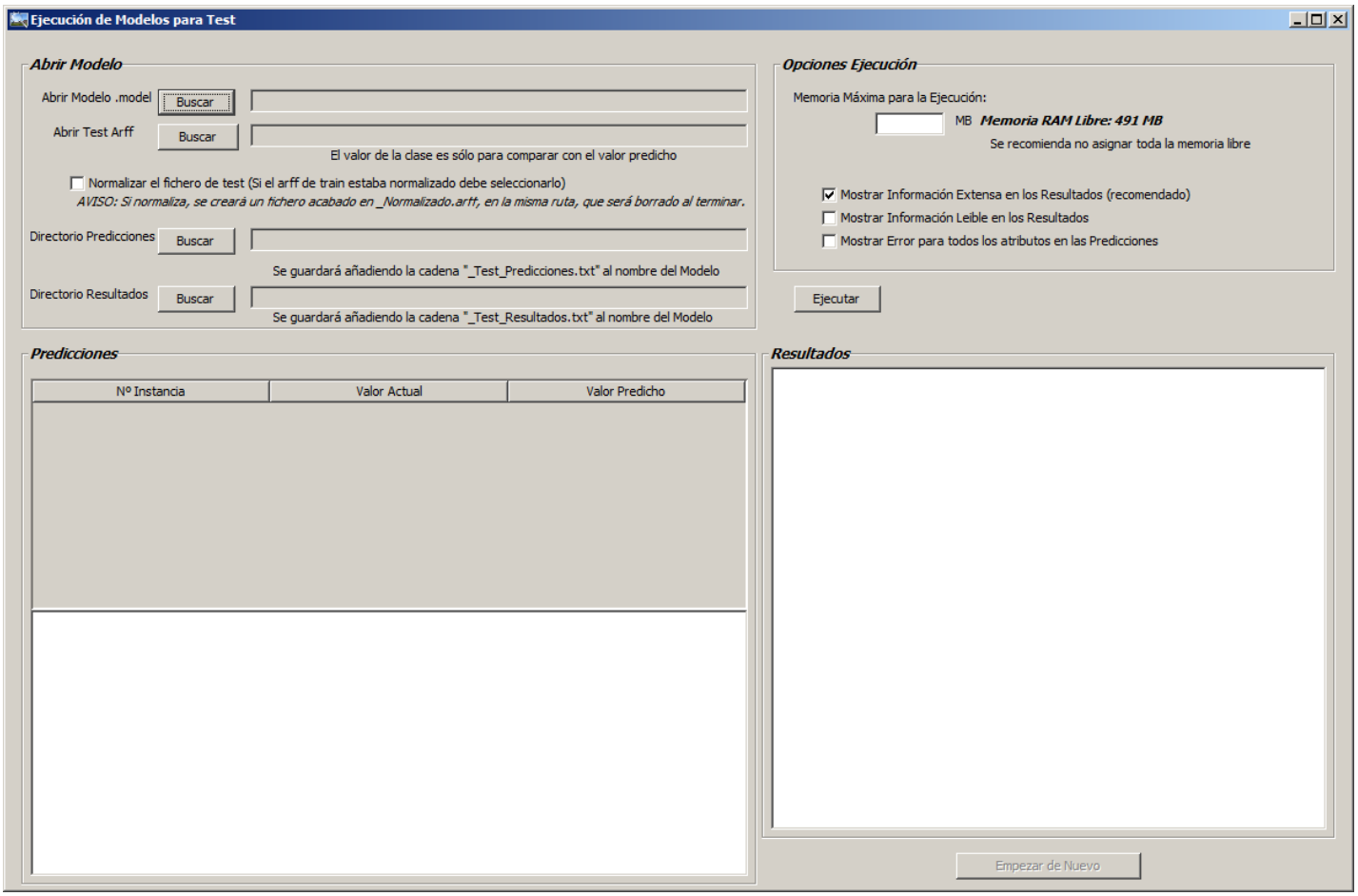

Figura 88: Ventana de la ejecución de modelos para test

Como se puede ver en la Figura 88, en primer lugar se abrirá el fichero con el modelo que se quiere ejecutar (formato *model*). También se abrirá el fichero *arff* con las instancias de test, seleccionando la casilla de normalizar, si durante la ejecución del experimento se normalizaron los atributos de las instancias. Se elegirá donde guardar los ficheros de texto con las predicciones y los resultados del test.

Por último se seleccionará algunas opciones de configuración, similares a las de la ejecución de experimentos para entrenamiento.

Se pulsará el botón de ejecutar y se esperará a que termine. Aquí no se podrá parar ni pausar ya que no tiene mucho sentido realizar estas operaciones debido al bajo tiempo de ejecución del modelo. En la Figura 89 se puede visualizar el resultado de una ejecución de un modelo.

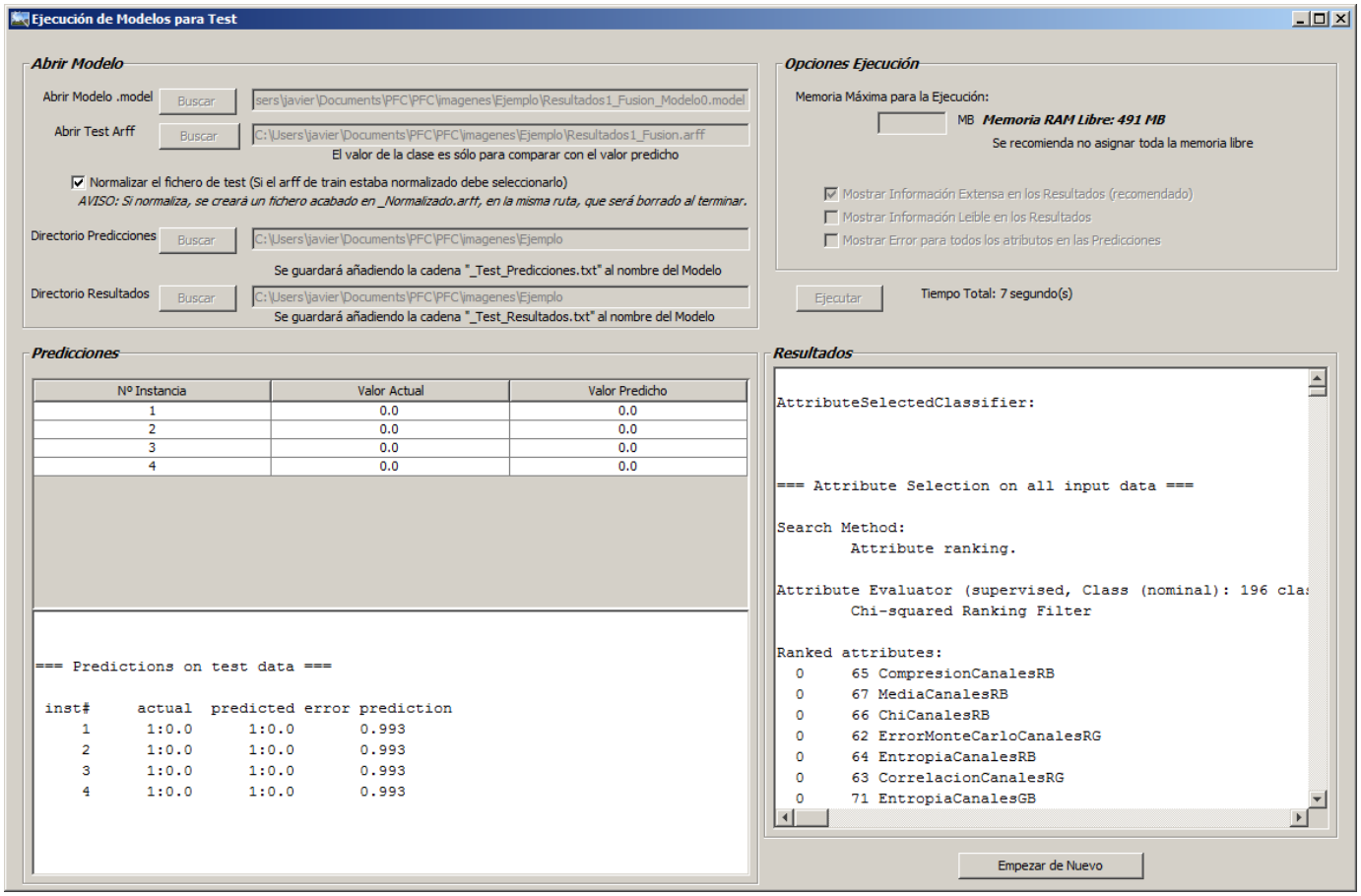

Figura 89: Ventana de la ejecución de modelos para test finalizada

En la Figura 89 se puede ver una tabla con la predicción hecha para cada nueva instancia. El valor actual, no influye en la ejecución, simplemente es el valor de la clase que tiene dada cada nueva instancia a evaluar, pudiendo ser relevante si está clase está bien otorgada, es decir, se sabe que esa clase es correcta para esa imagen y simplemente se quiere evaluar el modelo. Por lo que hay que fijarse en los resultados de la derecha, donde se verán todo tipo de errores y medidas comparativas. O también puede ocurrir, que se dé, a las nuevas instancias, una clase sin saber realmente si está bien dada, y por lo tanto, sólo interesará el valor de la predicción.

Cuando se finalice, se podrá realizar una nueva ejecución de un modelo pulsando sobre el botón de *Empezar de Nuevo.*

13. Analizar imágenes

Este módulo está separado del resto ya que su objetivo no es crear estegoanalizadores, si no analizar imágenes con un estegoanalizador que se creó a partir del mejor modelo de una serie de experimentos. La Figura 90 muestra la interfaz del analizador de imágenes:

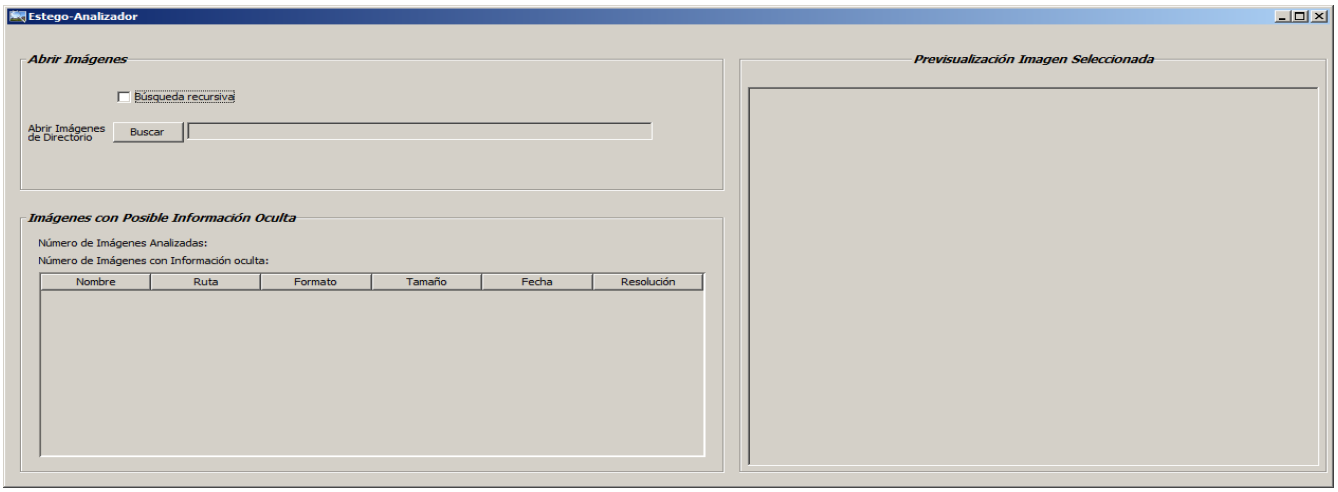

Figura 90: Ventana de analizador de imágenes

El análisis de imágenes funciona como cualquier otro estegoanalizador existente. Mediante el botón *Buscar* se podrá cargar el directorio, cuyas imágenes se quieren analizar. Si se desea analizar el contenido de los subdirectorios del directorio cargado, se deberá marcar la opción de *Búsqueda Recursiva.* Inmediatamente después de haber elegido un directorio, comenzarán a analizarse las imágenes de este.

Según el estegoanalizador encuentre imágenes estás se irán cargando en la tabla, pudiéndose pulsar en cada imagen para visualizarla en la parte de la derecha de la interfaz (Figura 91).

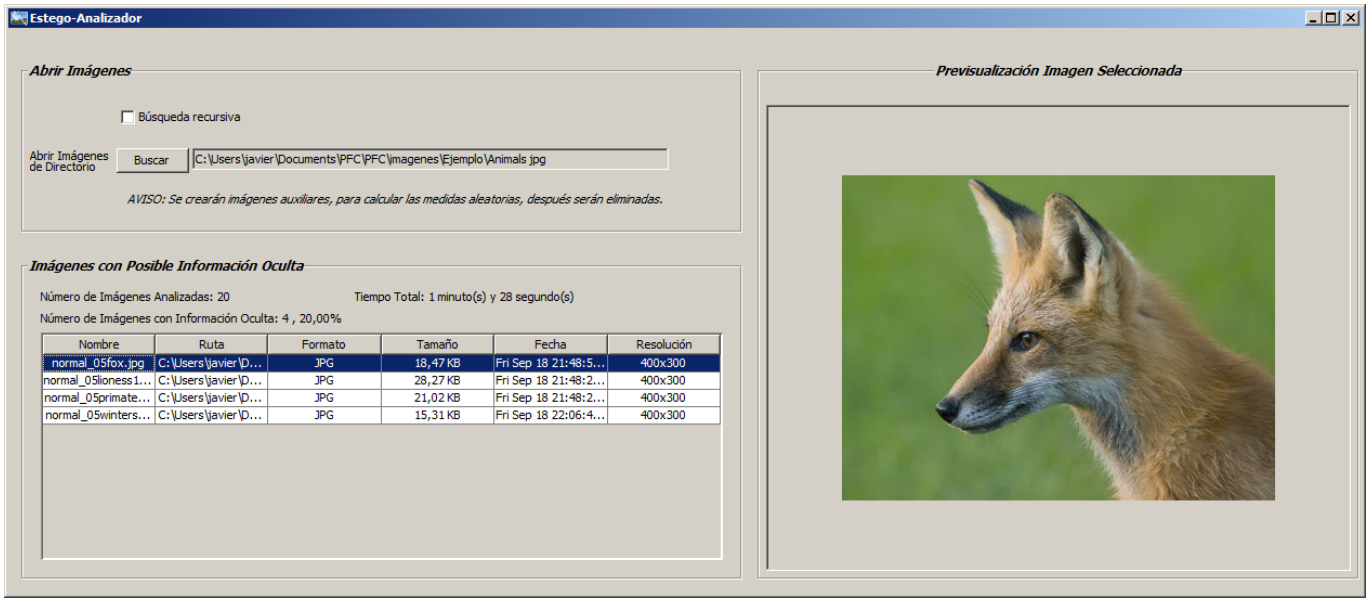

Figura 91: Ejecución del analizador de imágenes

Anexo 2. MANUAL DEL PROGRAMADOR

Para ampliar la herramienta, el programador tiene tres formas posibles de hacerlo:

- **Añadir Nuevas Técnicas Esteganográficas.**
- **Añadir Nuevas Medidas para su Extracción.**
- **Añadir Nuevos Clasificadores.**

A continuación, se va a explicar cómo cambiar el código para realizar ampliaciones en cada una de estos tres apartados. Es importante seguir en orden todo el proceso descrito.

Se recomienda usar el entorno de desarrollo *NetBeans IDE*, ya que es con el que se ha realizado *JUBSAC* y ofrece una serie de opciones que harán la modificación del código más sencilla.

1. Añadir Nuevas Técnicas Esteganográficas

Para añadir una nueva herramienta, esta debe ser ejecutable desde la línea de comandos de un terminal de Windows, siendo un archivo *exe* o un archivo *jar* o un archivo *bat***.** El paquete donde hay que realizar modificaciones es ***JUBSAC.aplicarEstegoProgramas*.**

1.1 Creación de la Ventana de Configuración de la Nueva Herramienta

Hay que crear una nueva interfaz de la herramienta que se quiera añadir para configurarla antes de la ejecución. La forma más sencilla de hacerlo es partiendo de una ya existente.

Para ahorrar esfuerzo, si la herramienta que se quiere añadir comparte los mismos formatos de imágenes de entrada que alguna de las ya existentes, se debería copiar esta.

Una vez está creada la nueva clase, lo primero que se debe hacer es modificar los elementos de la ventana, o Frame. Principalmente son el nombre de la ventana, las etiquetas o *labels* que informan de los detalles de la aplicación, que deberán ser cambiados a las opciones de la nueva herramienta.

1.2 Modificación del Código de la Nueva Herramienta

Después se procederá a modificar el código. Se usará como plantilla *Blindside.java***.** Ir al método ***jButtonGuardarXMLActionPerformed,*** y fijarse en la línea:

```
jfc.setSelectedFile(new File(new
Reemplazar().reemplazarUltimoPunto(this.ficheroOriginal.getAbsolutePath(),"_steg_Blindside."))));
```

Se cambiará la cadena Blinside por el nombre de la nueva herramienta. También en la siguiente linea:

```
jfc.setSelectedFile(new File(new
java.io.File(System.getProperty("user.dir")).getAbsolutePath().concat("\\Resultados1_steg_Blindside.
xml")));
```

Después ir al método ***jButtonTerminarActionPerformed***, fijándose en el siguiente bloque de código:

```
if (new LeerXmlConjunto().abrirCjto(this.nombreFicheroOriginal).getImagenesBMP().length == 0) {
                javax.swing.JOptionPane.showMessageDialog(this,
                        "Por favor, elija un conjunto con alguna imagen bmp.",
                        "Error",
                        javax.swing.JOptionPane.ERROR_MESSAGE);
}
```

En él se comprueba si el conjunto de entrada tiene alguna imagen de entre las que puede ejecutar el programa. Hay que modificar el nombre ***getImagenesBMP*** por uno que extraiga las imágenes en los formatos de la nueva herramienta. Y cambiar también el mensaje del error.

Hay varios métodos como este, dependiendo del formato:

- **getImagenesBMP**: extrae las imágenes en formato BMP.
- **getImagenesJPEGoJPG**: extrae las imágenes en formato JPG o JPEG.
- **getImagenesGIF**: extrae las imágenes en formato GIF.
- **getImagenesBMP_GIF_JPEGoJPG**: extrae las imágenes en formato BMP, GIF, y JPEG o JPG.
- **getImagenesBMP_JPEGoJPG:** extrae las imágenes en formato BMP, JPEG o JPG.
- **getImagenesPNG:** extrae las imágenes en formato PNG.

Si alguna combinación de formatos no estuviera aquí habría que realizar un método para esta. Para ello hay que ir a la clase ***JUBSAC.utiles.Conjunto.java.*** Lo más cómodo es copiar un método de los anteriores y modificar este:

```
if(this.imagenes[i].getName().endsWith(".png")|| this.imagenes[i].getName().endsWith(".PNG")){
```

Se modificarán las condiciones del if, para que estén todos los formatos posibles tanto en mayúscula como en minúscula.

1.3 Modificación del Código de JUBSAC

Una vez se ha modificado el código de la nueva herramienta se debe ir a ***JUBSAC.aplicarEstegoProgramas.EjecucionHerramientas.java,*** donde habrá que hacer una serie de ampliaciones para añadir la nueva herramienta.

La forma más sencilla es buscar el nombre de una herramienta base, como en este caso *Blindside,* ir copiando el código de esta e irlo adaptando a las nuevas necesidades. Se va a proceder a explicar qué pasos hay que realizar en ***EjecucionHerramientas.java:***

La primera modificación es sobre la interfaz gráfica. Sólo hay que fijarse en el resto de componentes de las herramientas, copiar estos y cambiarles el nombre. Cambiando el texto de estos, sólo si fuese necesario. Según la herramienta que se quiera añadir tendrá unos componentes u otros. Hay dos tipos:

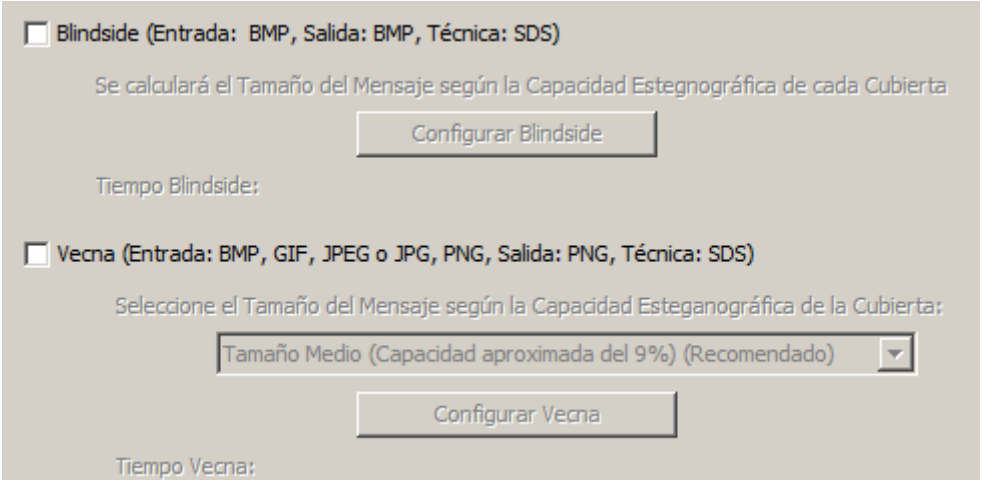

Figura 92: Tipos de componentes de las herramientas en EjecuciónHerramientas.java

Como se puede ver en la Figura 92, *Blinside* tiene unos componentes y *Vecna* otros. En Vecna se puede elegir el tamaño que se quiere ocultar en cada imagen, y en Blindside no. Esto es debido a que Blindside dispone de un parámetro que muestra cuantos bytes puede ocultar para cada imagen, y Blindside es ejecutado previamente con ese parámetro para calcular así el tamaño del mensaje secreto, que se ocultará para esa imagen. Si la herramienta que se quiere añadir no dispone de un parámetro que le permita predecir el tamaño del mensaje, se deberán copiar los componentes de Vecna, si no se copiarán los de Blindside. Si se dispone de este tipo de parámetro es más recomendable usarlo, porque puede ocurrir que el programa muestre error con alguna imagen si es demasiado grande el tamaño del mensaje a ocultar elegido.

Para añadir los nuevos componentes se deberá mover el resto de elementos para disponer de espacio suficiente. Una forma cómoda de trabajar es fijando el *layout* a *Null Layout*, mover los elementos y volver a seleccionar el *layout* a *Free Design*.

Una vez creadas los componentes de la interfaz se procederá a modificar el código para terminar de añadir la nueva herramienta, donde también se crearán los controles de los componentes.

Al principio de la clase se declaran unas variables globales:

```
OpenStego openStego = null;
Hide4PGP hide4PGP = null;
GifShuffle gifShuffle = null;
F5 f5 = null;
Steghide steghide = null;
Blindside blindside = null;
Vecna vecna = null;
```

Hay que añadir una igual para la clase recién creada. También hay otras variables globales para cada herramienta en concreto. Para Blindside son:

```
boolean soyElPrimero Blindside = false;
boolean soyElPrimeroTiempo Blindside = false;
long tiempoBlindside = 0;
```

Se añadirán unas similares pero para la nueva herramienta a añadir. Después se avanzará en el código hasta el método ***comprobarProgramas***. Donde existen una serie de ifs por cada herramienta y un if final donde se comprueba si el número de programas seleccionados y configurados es cero.

```
        if(this.jCheckBoxBlindside.isSelected()) {
            if(this.blindside!=null && this.blindside.estaTodoSeleccionado()){
                this.numProgramasSeleccionadosYConfigurados++;
            }else{
                return false;
            }
        }

        if(this.jCheckBoxVecna.isSelected()) {
            if(this.vecna!=null && this.vecna.estaTodoSeleccionado()){
                this.numProgramasSeleccionadosYConfigurados++;
            }else{
                return false;
            }
        }

        if(this.numProgramasSeleccionadosYConfigurados==0){
            return false;
        }else{
            return true;
        }
```

Se deberá añadir un if similar al de Blinside o Vecna, antes del último if donde se comprueba el número de programas. En cada if hay que escribir las variables correspondientes a la clase de la nueva herramienta.

Después continuar hasta el método ***habilitarBotonesBarra***. Donde se avanzará al final y se añadirá un bloque de líneas similares a las siguientes, pero para la nueva herramienta a añadir.

```
this.jCheckBoxBlindside.setEnabled(false);
this.jLabelSeleccioneBlindside.setEnabled(false);
this.jButtonConfigurarBlindside.setEnabled(false);
```

A continuación, por comodidad, situarse debajo del método ***calcularTiempoVecna*** y añadir un método similar a ***darValorVariablesVecna*** y otro similar a ***calcularTiempoVecna***. La forma más cómoda es copiar estos y después modificarlos.

Lo que se hará, simplemente es cambiar las palabas que contienen Vecna, por el nombre de la nueva herramienta. Por homogeneidad con el resto de código se recomienda referirse a la nueva herramienta en mayúscula, sólo la primera letra, ya que todas las herramientas empiezan de esta manera, menos cuando sea el nombre del objeto donde deberá ir en minúscula, como *vecna*.

El código de ***darValorVariablesVecna:***

```
private void darValorVariablesVecna(){
        if(soyElPrimeroVecna){
                    soyElPrimeroVecna = false
                    herramientaGlobal = "Vecna";
                    nombreFicheroOriginal = vecna.getNombreFicheroOriginal();
                    ficheroOriginal = vecna.getFicheroOriginal();
                    nombreFicheroDestino = vecna.getNombreFicheroDestino();
                    ficheroDestino = vecna.getFicheroDestino();
                    rutaGuardarImagenes = vecna.getRutaGuardarImagenes();
                    ficheroRutaGuardarImagenes = vecna.getFicheroRutaGuardarImagenes();

                    cDestino = new Conjunto();

                    cOrigen = l.abrirCjto(nombreFicheroOriginal);
                    imagenesOriginales = cOrigen.getImagenes();
cDestino.setNombre(ficheroDestino.getName().substring(0, ficheroDestino.getName().indexOf(".xml")));
                    cDestino.setFecha(new Date());
                    cDestino.setNumImagenes(imagenesOriginales.length);
                    imagenesFinales = new File[imagenesOriginales.length];

                    tamImagenesFinales = 0;
                    numEjecucion = -1;
                    numHilosVivos = numCores;
                    rutasImagenesDestino = new String[imagenesOriginales.length];
                    comandos = new String[imagenesOriginales.length];

                    tiempoVecna = new Date().getTime();
}
```

Se modificarán todas las variables acabadas en Vecna (***soyElPrimeroVecna, tiempoVecna, vecna,…***) por el nombre de la nueva herramienta, modificar también la cadena “Vecna”.

El último cambio importante en este método es en la siguiente línea:

```
imagenesOriginales = cOrigen.getImagenes();
```

Aquí se deberá reemplazar el nombre del método ***getImagenes*** por uno de los vistos anteriormente como ***getImagenesBMP, getImagenesGIF,…*** según los formatos de entrada que reconozca la nueva herramienta.

El código de ***calcularTiempoVecna:***

```
private void calcularTiempoVecna(){
        if(soyElPrimeroTiempoVecna){
            soyElPrimeroTiempoVecna = false;
            tiempoVecna = new Date().getTime() - tiempoVecna;
            jLabelTiempoVecna.setVisible(true);
            jLabelTiempoVecna.setText("Tiempo Total: " + convertirFecha(this.tiempoVecna));
        }
    }
```

Se deberá escribir el nombre de la nueva herramienta en cada variable acabada en Vecna, para así formar el nombre de las nuevas variables globales que se añadieron para usar esta.

Después, continuar avanzando en el código hasta encontrar la clase embebida ***MiThread***. La cual instanciará cada hilo paralelo en la ejecución. Hay que detenerse en su método ***run.*** En él hay un if por cada herramienta, ya que estas se irán ejecutando en orden, según estén

seleccionadas y configuradas en la interfaz. Avanzar al final del método, donde reutilizar algún if ya creado, para copiarlo y reeditarlo. Por ejemplo el último if de *Vecna*.

```
if(jCheckBoxVecna.isSelected() && !estamosParados){
            synchronized(this.padre)
            {
                darValorVariablesVecna();
            }
            ejecucionHilo("Vecna", padre);
            synchronized(this.padre)
            {
                calcularTiempoVecna();
            }
        }
```

Cambiar el nombre de *Vecna* por el de la nueva herramienta. También cambiar la cadena *"Vecna"*, esto es el nombre de la herramienta, que actúa como identificador a la hora de la ejecución.

Continuar en el código hasta ***ejecucionHilo***, hasta donde hay unos if-else encadenados para cada herramienta:

```
if(herramientaParticular.compareTo("OpenStego")==0){
                                ……

}else if(herramientaParticular.compareTo("Hide4PGP")==0){
                                ……
```

Estos ifs contienen la creación de un comando que será ejecutado más tarde, por lo que es importante que esté bien construido y esto depende de la herramienta nueva que se esté añadiendo, pudiendo no compartir los mismos detalles que las herramientas que ya están implementadas en JUBSAC.

Se debe añadir un else if al final del bloque de if-else, pero dentro de este bloque. También se debe prestar atención a las diferencias entre el contenido de cada if, porque depende de la herramienta que se ejecute. Básicamente hay dos tipos, los vistos anteriormente, según se pueda, con la herramienta, calcular cuánto ocupa el mensaje o no. Para explicar el código de cada uno de los dos casos se usará el código de *OpenStego* y *GifShuffle**.***

```
if(herramientaParticular.compareTo("OpenStego")==0){
  rutasImagenesDestino[ejecucionParticular] = rutasImagenesDestino[ejecucionParticular]+".png";
  numBytesSecreto = es.escribirSecretoElegido(jComboBoxTamanoOpenStego.getSelectedIndex(),
                      imagenesOriginales[ejecucionParticular],
                      secretos[Integer.parseInt(Thread.currentThread().getName())],
                      herramientaParticular);
  comandos[ejecucionParticular] = "java -jar StegoProgramas\\openstego.jar --embed --algorithm=lsb –
        messagefile="+"\""+secretos[Integer.parseInt(Thread.currentThread().getName())]+"\""+" --
        coverfile="+rutaImagenOriginal+" --stegofile="+"\""+rutasImagenesDestino[ejecucionParticular]+"\""+"
        --compress --encrypt --password "+password;

}
```

Se puede ver el código del if de *OpenStego**,*** donde en la primera línea se crea la ruta de la imagen de destino, y esta va formada por *png* al final, ya que el formato de salida de todas las

imágenes de *OpenStego* son *png*. Esta línea sólo deberá usarse cuando la nueva herramienta sólo tenga un único formato de salida y pueda no ser el mismo de la imagen de entrada.

En la siguiente línea se calcula el número de bytes que debe ocupar el mensaje secreto en función de la opción elegida por el usuario en el *Combo Box*, ***jComboBoxTamanoOpenStego.getSelectedIndex()***. Se usa ***es,*** un objeto de la clase ***EscribirSecreto***, el cual es usado para escribir el fichero TXT con el secreto aleatorio en función del tamaño elegido por el usuario.

Para todas las líneas, que se usen en la nueva herramienta, se deberá ir cambiando el nombre de las variables de *OpenStego* por el de la nueva herramienta. La última linea es el comando de ejecución de la herramienta. La herramienta deberá estar dentro de la carpeta ***StegoProgramas***. Si la herramienta no es un único ejecutable (exe, java o bat) y necesita más archivos como librerías, se recomienda que se cree una carpeta para ubicarlas. Para escribir el comando se recomienda haber probado este antes desde un terminal para saber que funciona correctamente. Los programas esteganográficos suelen necesitar tres parámetros:

- Mensaje secreto: ***secretos[].***
- Cubierta: ***rutaImagenOriginal***.
- Estego-Objeto: ***rutasImagenesDestino[].***

Estas variables deberán ser ubicadas, como se realiza con *OpenStego*, en el comando, en el sitio correcto que necesite el comando de la nueva herramienta. Además si la nueva herramienta dispone de más opciones como cifrado suele necesitar un parámetro más:

- Clave: ***password.***

Además, si hay opciones de compresión, se recomienda usarlas para hacer más complicado y seguro el resultado de la ocultación.

Otro detalle a tener en cuenta es que puede ocurrir que la herramienta no necesite un parámetro para el estego-objeto y lo que haga sea sobrescribir la cubierta. Para ello hay que hacer una copia del original y sobrescribir la copia:

```
try {
     rutaImagenOriginal = imagenesOriginales[ejecucionParticular].getAbsolutePath
     new ManejadorFicheros().copyFiles(rutaImagenOriginal, rutasImagenesDestino[ejecucionParticular]);
} catch (IOException ex) {
    Logger.getLogger(EjecucionHerramientas.class.getName()).log(Level.SEVERE, null, ex);
}
```

Para ello usar la clase ***JUBSAC.utiles.ManejadorFicheros***, la cual permite copiar un fichero en otro. Esto ocurre en el if-else de ***Hide4PGP.*** A continuación se continúa con el if-else de *GifShuffle:*

```
else if(herramientaParticular.compareTo("GifShuffle")==0){
    numBytesSecreto = cs.calcularTamanoSecreto("cmd /c start /b StegoProgramas\\gifshuffle.exe -S
                      "+"\""+imagenesOriginales[ejecucionParticular].getAbsolutePath()+"\"", "GifShuffle");
    es.escribirSecreto(numBytesSecreto,secretos[Integer.parseInt(Thread.currentThread().getName())]);
    comandos[ejecucionParticular] = "StegoProgramas\\gifshuffle.exe -p "+password+" -f
    "+"\""+secretos[Integer.parseInt(Thread.currentThread().getName())]+"\""+" "+ rutaImagenOriginal +"
    "+"\""+rutasImagenesDestino[ejecucionParticular]+"\"";
}
```

En este if-else, también se usa la clase ***EscribirSecreto***, pero antes calcula el tamaño del mensaje secreto según la imagen original que se esté procesando. Esto lo hace gracias a un objeto, ***"cs"***, de la clase ***JUBSAC.aplicarStegoProgramas.CalcularSecreto.java.*** En esta clase se ejecuta el comando de *GifShuffle* que calcula, o predice, cuanto espacio de la imagen puede usar como máximo para ocultar información. Este comando suele necesitar como único parámetro la imagen original o cubierta.

Si la nueva herramienta es de este tipo se deberá entonces, editar el método ***procesarSalida*** de ***CalcularSecreto.java.***

```
if(stegoPrograma.compareTo("GifShuffle")==0){
                                ……
}else if(stegoPrograma.compareTo("Steghide")==0){
```

Hay varios if-else, se deberá añadir uno para la nueva herramienta. Aquí se devuelve un número entero, el número de bytes que devuelve la salida de la ejecución del comando que se le pasa a esta clase. Por ejemplo para *GifShuffle* el comando devuelve la siguiente cadena al haberlo ejecutado:

```
GiffShuffle: File has storage capacity of 1683 bits (210 bytes)
```

Entonces en ***procesarSalida*** se deberá leer este String y extraer el número de bytes. Dependiendo de la herramienta esto puede ser más sencillo o menos, ya que por ejemplo hay herramientas que devuelven el valor en bytes o en kilobytes, entonces habrá que convertirlo a bytes y realizar la programación previniendo estos casos.

También es posible que el programa prediga mal cuanto tamaño tiene como máximo y a veces no sea suficiente, con lo que se devolverá un porcentaje menor para que la herramienta no falle. Entonces programar el if-else del método ***ejecuciónHilo***, fijándose en las herramientas ya programadas, pero teniendo en cuenta cómo se comporta la nueva herramienta a añadir. Por último explicar que en el if-else de *Vecna* hay dos tipos de comandos de ejecución de Java.

```
comandos[ejecucionParticular] = "java -Xmx128M -jar StegoProgramas\\vecna_0.6.jar -e -i ….
comandos[ejecucionParticular] = "java -jar StegoProgramas\\vecna_0.6.jar -e -i ….
```

Uno con 128 MegaBytes y otro sin ellos, esto es así porque para las imágenes de resoluciones altas *Vecna* necesita más memoria de la JVM para ejecutarse correctamente. Después ir hasta el método ***paralelizarProcesos*** de **EjecucionHerramientas.java**.

```
this.soyElPrimeroVecna = true;
this.soyElPrimeroTiempoVecna = true;
```

Donde se añadirán estas dos líneas debajo de las mismas, con el nombre de las variables de la nueva herramienta. Continuar hasta llegar al método ***sacarNumImagenesFinalesTotales***, donde hay varios ifs, uno por cada herramienta, y avanzar hasta el final del método:

```
if(jCheckBoxVecna.isSelected()){
    totales += (new
        LeerXmlConjunto().abrirCjto(vecna.getNombreFicheroOriginal())).getImagenes().length;
}
```

Añadir este if, pero para la nueva herramienta en vez de *Vecna*.

Después crear un método que implemente la funcionalidad del botón de *Configurar* la nueva herramienta. En *NetBeans* una forma muy cómoda es hacer doble clic sobre el botón, desde la parte de *Design* y creará la signatura del método directamente.

```
private void jButtonConfigurarHide4PGPActionPerformed(java.awt.event.ActionEvent evt) {
        this.setVisible(false);
        this.hide4PGP = new Hide4PGP(this);
        this.hide4PGP.setVisible(true);
}
```

Este método oculta la ventana de ejecución de herramientas esteganográficas y abrirá la propia de la herramienta, en este caso es *Hide4PGP*. Cambiar el nombre de esta herramienta por el de la nueva.

También configurar lo que sucede cuando se marca en la interfaz, el *Check Box* de la herramienta. Y también que ocurre cuando se desmarca.

```
private void jCheckBoxOpenStegoActionPerformed(java.awt.event.ActionEvent evt) {
        if(this.jCheckBoxOpenStego.isSelected()){//si la marco
            this.jButtonConfigurarOpenStego.setEnabled(true);
            this.jLabelSeleccioneOpenStego.setEnabled(true);
            this.jComboBoxTamanoOpenStego.setEnabled(true);
        }else{//si lo desmarco
            this.jButtonConfigurarOpenStego.setEnabled(false);
            this.jLabelSeleccioneOpenStego.setEnabled(false);
           this.jComboBoxTamanoOpenStego.setEnabled(false);
        }
    }
```

Cambiar *OpenStego* por el nombre de la nueva herramienta y si la herramienta no tiene *Combo Box* se eliminará la línea donde se habilita.

Ya finalizando avanzar hasta el método ***jButtonEmpezarDeNuevoActionPerformed.*** Aquí se reinician los valores de las variables globales, y del estado de la interfaz.

```
        this.jCheckBoxVecna.setEnabled(true);
        this.jCheckBoxVecna.setSelected(false);
        this.jLabelSeleccioneVecna.setEnabled(false);
        this.jComboBoxTamanoVecna.setEnabled(false);
        this.jComboBoxTamanoVecna.setSelectedIndex(1);
        this.jButtonConfigurarVecna.setEnabled(false);
        this.jLabelTiempoVecna.setText("Tiempo Total:");
```

Avanzar hasta el bloque de código anterior, donde se copiará y modificará debajo para la nueva herramienta. Si la herramienta no tiene *Combo Box* se eliminarán las líneas en las que se deshabilita y se marca una selección de este.

2. Añadir Nuevas Medidas para su Extracción

Para añadir nuevas medidas, situarse en el paquete ***JUBSAC.extraerMedidas***. La clase principal de este paquete es ***ExtraerMedidas,*** donde se encuentra la GUI.

Si se abre esta con *NetBeans* se podrá editar directamente la interfaz y añadir aquí las nuevas medidas.

Si las nuevas medidas están incluidas en alguno de los cuatro grupos existentes, y hay espacio suficiente en la pestaña donde estas se encuentran, se podrán añadir en la pestaña de su tipo:

- **Aleatoriedad.**
- **Estadísticos.**
- **Estegoanálisis.**
- **Características Imagen.**

Si no, es recomendable crear una pestaña nueva, añadiendo un *Tabbed Pane* y dentro de este un *JPanel*, con *Layout* a *Null Layout,* para más comodidad.

Una parte tediosa de la extracción de medidas, es que hay que tener en cuenta muchas acciones con los *Check box* que se añadan, ya que si por ejemplo se desmarca uno, referente a un tipo de medidas, todos los subtipos de este se tendrán que deshabilitar. Para esto es conveniente fijarse en métodos como ***jCheckBoxMedidasAleatoriasActionPerformed,*** donde se tiene en cuenta cada subtipo de este, y los subtipos de estos. Por ejemplo, se va a usar una comparación con el parentesco abuelo, padre e hijo. Si se tiene un *check box*, como ***jCheckBoxMedidasAleatorias,*** este será el abuelo ya que hay dos check box descencientes suyos como por ejemplo, el padre ***jCheckBoxMedidasAleatoriasImagenya*** y el hijo ***jCheckBoxEntropiaImagen.*** Si se desmarca el padre, el hijo se deshabilita, pero el abuelo no. Y si se desmarca el abuelo se deshabilitan el padre y el hijo. Pero si se deshabilita el hijo, y después el abuelo, cuando se habilite el abuelo el hijo deberá seguir deshabilitado ya que antes estaba marcado por el usuario. Esta comparación puede no parecer seria, pero para que la interfaz no tenga incoherencias hay que tener cuidado con todos estos detalles de control.

Una vez añadidos los *check box* de las nuevas medidas y los controles de estas ir al método ***ejecucionHilo***. Aquí hay un bucle para todas las imágenes del conjunto *XML* del que se va a extraer las medidas seleccionadas.

```
for(int i=0; i< imagenesOriginales.length && !estamosParados; i++){
```

Dentro hay un if por cada *check box* que hay en la interfaz. Estos se encadenan según los tipos y subtipos de medidas que haya.

```
if (this.jCheckBoxMedidasAleatorias.isSelected()){

  if (this.jCheckBoxMedidasAleatoriasImagen.isSelected()){
```

Si están seleccionados los *check box* de un tipo y los de sus subtipos se calculará la medida que corresponda, dentro del if, si está seleccionada.

Se recomienda que se calculen las medidas en otras clases, para practicar una buena programación orientada a objetos. La forma de calcular la nueva medida depende de esta. En las medidas existentes se calculan de varias formas:

1. Usando un programa **ejecutable** externo.
2. Usando código de **librerías** externas.
3. Usando **código** propio.

Se tiene que:

1. Para la primera forma se usa el programa *ENT.exe* que calcula unos tests estadísticos. A este programa hay que proporcionarle un archivo, como entrada, y como salida da el resultado de aplicar los tests a este archivo. En algunos casos la imagen que se quiere calcular tiene que ser modificada. Como sucede cuando se separa en canales de color RGB o porciones. En este caso se crea una nueva imagen que después de haber sido ejecutada por ENT, es eliminada.
2. Para la segunda forma, se usan librerías de DIIT, para calcular análisis estadísticos.
3. Y para la tercera forma, se usa código propio, se usan otras clases creadas, que implementan fórmulas para calcular medidas estadísticas o una tasa estegoanalítica.

Cuando se ha terminado de calcular la medida lo que siempre se ejecuta es lo siguiente:

```
cabecera.append("@attribute MediaImagen real"+"\n");
```

Esto añade un nuevo atributo al *arff* final. Dar a este un nombre breve, pero identificativo, y único, q no se repita de entre todas las medidas.

```
auxPatron[numAtributos] = propiedadesAleatoriasImagen.getMedia();
numAtributos++;
```

auxPatron se igualará siempre al valor de la nueva medida calculada, dependiendo de cómo haya sido calculada. Y se aumentará el número de atributos del *arff*.

A continuación, en ***resetearVariablesGlobales*** hay que reinicializar el valor de cada nueva variable añadida, para cuando el usuario pulse el botón de *Empezar de Nuevo*. Variables añadidas serán, por ejemplo, las nuevas clases que calculen las nuevas medidas.

Finalmente, habrá que añadir en los métodos ***habilitarTodo***, ***deshabilitarTodo***, y ***seleccionarTodo***, los nuevos *Check Box* añadidos en la interfaz para habilitarlos, deshabilitarlos y seleccionarlos cuando sea necesario.

3. Añadir Nuevos Clasificadores

Los clasificadores usados por ***JUBSAC*** tienen que poder ser ejecutados desde la herramienta Weka. Para añadir nuevos clasificadores hay que trabajar en el paquete ***JUBSAC.experimentador***. Donde se abrirá primero ***CreacionExperimentos.java.*** Cargar la interfaz, para añadir al menú desplegable el nuevo clasificador.

Figura 93: Menú desplegable de clasificadores en CreacionExperimentos.java

Este menú se llama ***jComboBoxClasificador.*** Tiene una propiedad llamada ***model***, donde se encuentra una lista de todos los clasificadores permitidos. Aquí se añadirá el nuevo nombre del clasificador. El siguiente paso es ir a la clase ***EjecucionExperimentos.java,*** hasta el método ***paqueteWeka***:

```
public String paqueteWeka(String nombre){
        if(nombre.compareTo("NaiveBayes")==0){
            return "weka.classifiers.bayes."+nombre;
        }else if(nombre.compareTo("Logistic")==0 || nombre.compareTo("MultilayerPerceptron")==0
                || nombre.compareTo("RBFNetwork")==0 || nombre.compareTo("SMO")==0
                || nombre.compareTo("VotedPerceptron")==0){
            return "weka.classifiers.functions."+nombre;
        }else if(nombre.compareTo("IB1")==0 || nombre.compareTo("IBk")==0
                || nombre.compareTo("KStar")==0 || nombre.compareTo("LWL")==0){
            return "weka.classifiers.lazy."+nombre;
        }else if(nombre.compareTo("J48")==0 || nombre.compareTo("REPTree")==0){
            return "weka.classifiers.trees."+nombre;
        }
        return null;
    }
```

Este método devuelve la cadena del paquete de Weka donde está el clasificador a ejecutar. Según el nombre que se haya añadido a ***jComboBoxClasificador*** habrá que usar un if-else existente, si coincide que el clasificador añadido está en un paquete existente. O crear un if-else nuevo si no existe el paquete y devolver en este el paquete entero con el clasificador al final. El algoritmo añadido debe ser un clasificador válido para el tipo de atributos generados por la extracción de medidas.Finalmente, en el mismo modelo, situarse en la clasee, ***LeerModelo***. En ella hay un método llamado ***buscarClasificador.*** Este método recorrerá un fichero *model* para buscar el clasificador que pertenece a este. Por lo tanto, si se quiere usar un nuevo clasificador, tendrá que añadirse a este método.

```
                if(linea.contains("weka.classifiers.bayes.NaiveBayes")){
                    return "weka.classifiers.bayes.NaiveBayes";
                }
                if(linea.contains("weka.classifiers.functions.Logistic")){
                    return "weka.classifiers.functions.Logistic";
                }……
```

Son una serie de ifs, por cada clasificador, donde se devolverá el paquete entero incluyendo el nombre del clasificador. Se deberá añadir un nuevo if para el nuevo clasificador.